UNIVERSIDAD CARLOS III DE MADRID
ESCUELA POLITÉCNICA SUPERIOR

DEPARTAMENTO DE INFORMÁTICA

INGENIERÍA DE TELECOMUNICACIÓN

PROYECTO FIN DE CARRERA

Geolocalización en Tiempo Real para Dispositivos Móviles: Localización en Interiores.

Autor: Fernando García Radigales
Tutor: Jorge Blasco Alís

Leganés, 27 de Julio de 2012

Agradecimientos

Esta es sin duda la parte más difícil de escribir, cómo concentrar tanto en tan pocas líneas. Sin embargo no podía dejar pasar la oportunidad de aprovechar esta ocasión tan especial, que supone el fin de una etapa muy importante en mi vida, para mostrar mi más sincero agradecimiento a todas las personas que han estado a mi lado todos estos años.

Ante todo quiero dar las gracias a mi familia, a mis padres Juan Antonio y Rosario y a mi hermano Jorge, que son los cimientos de mi vida y me han brindado la posibilidad de recibir esta educación. Quiero agradeceros vuestra paciencia, dedicación y confianza en mí, vuestro incondicional apoyo y el estímulo constante de vuestras palabras, que han servido de guía y motivación desde antes de comenzar esta carrera.

A quienes habéis sido compañeros y ahora amigos, alguno ya lo erais antes de empezar, con vosotros he compartido esfuerzos, sufrimientos y éxitos y tantos viajes, cenas, risas y otros momentos imborrables dentro y fuera de las aulas. Especialmente a Natalia, más que compañera y amiga, por los años que me acompañaste en cada paso.

Parte del éxito de este proyecto se lo debo a mi socio Francesco, juntos dimos forma de PFCs a lo que solo era una idea y a Blas, mi Tutor, quien ha sabido con su experiencia explotar todo el potencial y transformarlo en un proyecto real.

Y para finalizar, por supuesto, no podía faltar Silvia, has llegado a mi vida al final de esta etapa, pero este último año has estado a mi lado compartiendo mis retos y mis logros, me has contagiado de tu energía y me has dado la fuerza y la confianza para llegar hasta el final. Contigo cierro esta etapa de mi vida y comienzo la siguiente, cargada de nuevas metas e ilusiones.

Resumen

El presente proyecto titulado *"Geolocalización en Tiempo Real para Dispositivos Móviles: Localización en Interiores."* afronta el diseño de una aplicación para dispositivos móviles *Android* que permita obtener la localización de un usuario y compartirla con sus contactos sobre un mapa. La aplicación incluye una serie de funcionalidades y características que la diferencian de aplicaciones existentes.

La principal es el funcionamiento en cualquier entorno, tanto exterior, dónde se dispone de señal GPS, como interior, dónde al no contar con señal GPS se apoyará en un sistema de localización basado en puntos de acceso WIFI. El diseño del sistema de localización, partiendo del estudio de los parámetros de las señales electromagnéticas, del análisis de los métodos de localización, su posible aplicación y particularización para este proyecto también se aborda en este documento.

Otra de las características destacadas es el funcionamiento en tiempo real. Se diseñará el sistema de forma que los resultados de geolocalización se obtengan bajo demanda de los usuarios, garantizando la privacidad y evitando la necesidad de constante actualización de esta información, con las ventajas que ello supone.

Por último, la aplicación podrá incorporar mapas personalizados, dotándola de una gran utilidad en escenarios como por ejemplo las pistas de una estación de esquí, donde los planos convencionales de calles o caminos no se ajustan a las ubicaciones reales de los usuarios.

Palabras clave: Geolocalización, localización en interiores, GPS, WiFi, tiempo real, mapas personalizados, Android, C2DM, RSSI.

Índice

Índice de Figuras

Índice de Requisitos y Procesos

Índice de Pruebas

Índice de Tablas

Capítulo 1: Introducción

1.1 Motivación

Los teléfonos móviles han revolucionado completamente las comunicaciones. La facilidad con que permiten transmitir información por voz o texto de forma inalámbrica, ha hecho posible la interacción entre personas en cualquier momento.

Con el avance de la tecnología, los teléfonos móviles ofrecen más y mejoradas características de funcionamiento, procesamiento y comunicación. La aparición de los *smartphones* o teléfonos inteligentes permite remplazar a los ordenadores personales en algunos casos.

El éxito de los *smartphones* reside en sus avanzadas características y en la capacidad de instalar diferentes aplicaciones específicamente diseñadas que amplían las funcionalidades. Las aplicaciones móviles han revolucionado la forma en que interactuamos con el mundo, son programas que se pueden descargar en el teléfono y que aportan innumerables posibilidades: participar en juegos, obtener indicaciones de localización, acceder a noticias, libros, datos del tiempo, etc.

Las aplicaciones se obtienen a través de la plataforma propia de cada sistema operativo, en las que los desarrolladores las ponen a disposición de los usuarios. Los desarrolladores tienen a su alcance las herramientas que posibilitan programar sus propias aplicaciones y ofrecerlas en estas plataformas al resto de usuarios.

En España, según cifras de Abril de 2012 publicadas por la CMT [1], el número de líneas de telefonía móvil para comunicación personal era de unos 55.245.981, siendo la tasa de penetración de 119,6 líneas por cada 100 habitantes.

El cuantioso número de *smartphones* y sus avanzadas características junto la accesibilidad al desarrollo de aplicaciones, hacen del mercado de aplicaciones móviles un atractivo sector en el que cada día más empresas vuelcan su actividad. Si bien es un mercado que se considera ya maduro, tiene aún un largo recorrido y margen de crecimiento.

Existen millones de aplicaciones móviles, desde las más sencillas hasta las que realizan tareas más específicas y cada día aparecen cientos de nuevas aplicaciones. Pero no está todo inventado, sigue habiendo posibilidad de crear aplicaciones innovadoras y que cubran nuevas necesidades.

El conocimiento de la ubicación, es una de las capacidades más relevantes de los dispositivos móviles y las aplicaciones basadas en la posición están ganando cada vez más importancia en la industria y entre los usuarios. Son comunes las aplicaciones que permiten a los usuarios obtener su posición actual, gracias a tecnologías como el GPS integradas en los dispositivos

móviles. Además estas aplicaciones permiten compartir la posición obtenida con otros usuarios sobre un mapa.

Este proyecto surge de una necesidad real. Si bien existen aplicaciones comerciales con estas funcionalidades de posicionamiento o localización, algunas de ellas ampliamente extendidas, se han detectado ciertas limitaciones. En situaciones específicas, con problemas de cobertura GPS, con requerimientos de tasas de actualización altas o donde los mapas de que disponen no se ajustan a la realidad, estas aplicaciones pierden su utilidad.

Una de estas limitaciones se produce en recintos interiores, en los que obtener la posición GPS no resulta posible. Con un sistema capaz de detectar la posición del usuario en este tipo de escenarios, las aplicaciones de localización podrían seguir siendo de gran utilidad para los usuarios.

1.2 Alcance del proyecto

Las tareas necesarias para realizar una aplicación de las características de la que se pretende construir, se dividen dada su magnitud, en dos proyectos estrechamente ligados. El otro proyecto, cuyo autor es **Francesco Gatti Gómez**, parte de algunas decisiones enunciadas en este proyecto e integra parte de los resultados aquí obtenidos.

A continuación se detallan las responsabilidades de cada proyecto:

PFC Fernando García Radigales:

El proyecto que describe este documento trata el análisis de negocio de la aplicación a desarrollar, analiza las tecnologías necesarias y define la arquitectura, enfocándose en las características diferenciadoras de sus competidores y su viabilidad técnica. Se describe también el diseño, prueba e integración de un sistema de posicionamiento para recintos interiores.

Para el diseño de este sistema de posicionamiento, se desarrollarán aplicaciones de experimentación y se llevarán a cabo varios experimentos reales. Los resultados de estos experimentos determinarán parte de la lógica de la aplicación final.

PFC Francesco Gatti Gómez:

En este proyecto que nace del anteriormente mencionado, se detallan a nivel técnico los requisitos funcionales y no funcionales de la aplicación y su diseño. Se incluye en este proyecto la programación de los diferentes subsistemas (cliente y servidor) y el diseño del protocolo de comunicación entre ambos.

También integra parte de los resultados obtenidos sobre el diseño del sistema de posicionamiento en interiores y las pruebas de la aplicación final.

En el siguiente gráfico se aprecian las distintas tareas llevadas a cabo en ambos proyectos. El contenido recogido dentro del recuadro corresponde al proyecto descrito en este documento.

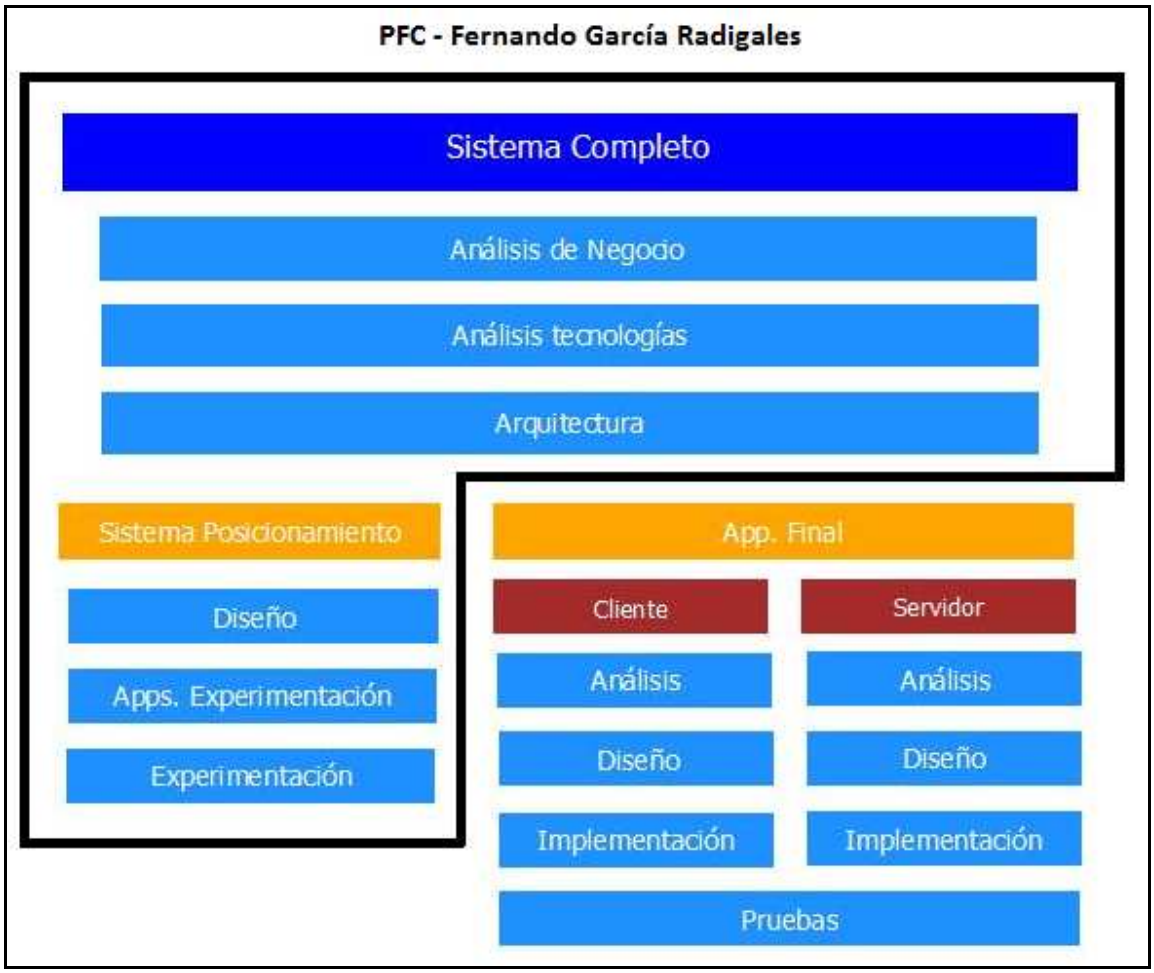

Figura 1: Distribución de tareas en proyectos

1.3 Objetivos

El objetivo final es desarrollar una aplicación para dispositivos móviles que permita a su usuario obtener y compartir su posición, obtener las posiciones de sus *contactos*[1] y mostrarlas sobre un mapa.

[1] Ver definición en Glosario

Se pretende aprovechar todas las características de los *smartphones*, para solventar las carencias que presentan otras aplicaciones con funcionalidades parecidas y así cubrir las necesidades de los usuarios en situaciones específicas, en las que una aplicación como la que se describe puede resultar de gran utilidad.

Para entornos en los que las condiciones de funcionamiento no son óptimas, como es el caso de recintos interiores, se desarrollará un sistema de posicionamiento, que permita el correcto funcionamiento de la aplicación.

Para este proyecto se definen los siguientes objetivos:

- Realizar un análisis de las aplicaciones móviles de geolocalización existentes, que permiten compartir la localización entre usuarios, identificando funcionalidades que no ofrecen.
- Diseñar la arquitectura de un sistema que permita conocer las posiciones de dispositivos móviles en tiempo real usando mapas personalizados.
- Desarrollar un sistema de posicionamiento en interiores capaz de obtener las posiciones de los usuarios con niveles aceptables de precisión.
- Realizar pruebas de campo que permitan alcanzar unos niveles de precisión aceptables del sistema de localización, desarrollando las herramientas necesarias para realizar dichas pruebas.

1.4 Contenido de la memoria

Se describe a continuación la estructura de este documento y el contenido de cada capítulo. En la primera parte de la memoria se analizará la aplicación a alto nivel, las funciones básicas y los procesos que debe realizar. Se detallarán las diferencias con las aplicaciones existentes y como se van a materializar. Se estudiarán las tecnologías necesarias para conseguir todas las características deseadas y se definirá la arquitectura del sistema.

- **Capítulo 1:** Capítulo introductorio en el que se describen las motivaciones del proyecto, el alcance y los objetivos. También se presentan el contenido de la memoria.
- **Capítulo 2:** Análisis de la Aplicación. Capítulo que detalla las funcionalidades requeridas para la aplicación final, los procesos y la arquitectura del sistema completo.
- **Capítulo 3:** Análisis Tecnológico. Se analizan las diferentes tecnologías y productos que nos permitirán desarrollar el proyecto y construir cada uno de los diferentes módulos.
- **Capítulo 4:** Características Diferenciadoras. En este capítulo se analizan algunas aplicaciones existentes y se ven las características que diferencian a la aplicación de sus competidoras.

La segunda parte del documento se centrará en el sistema de posicionamiento para recintos interiores, analizando las distintas alternativas para el diseño y las decisiones tomadas. Se implementará un conjunto de aplicaciones que permitirán realizar experimentos para este sistema, se llevarán a cabo los experimentos y se analizarán sus resultados.

- **Capítulo 5:** Sistema de Posicionamiento. Se estudia en profundidad en este capítulo, el sistema de posicionamiento para interiores, se ven algunas soluciones comerciales y se analizan las diferentes técnicas, eligiendo y adaptando la que más se ajuste a la aplicación.
- **Capítulo 6:** Aplicaciones para Experimentación. Se analizan y diseñan un conjunto de aplicaciones para obtener la información necesaria que permita tomar decisiones sobre las características del sistema de posicionamiento.
- **Capítulo 7:** Experimentación y Resultados. Se explican los experimentos y pruebas realizadas, las conclusiones obtenidas y la repercusión en la aplicación.

En la última parte del documento se explicará como se ha gestionado el proyecto, las conclusiones obtenidas y los trabajos futuros a desarrollar. También se incluye un glosario de términos que ayude a comprender todos los términos empleados en el proyecto y la bibliografía utilizada.

- **Capítulo 8:** Gestión del Proyecto. Se detalla la planificación del proyecto, las fases, los materiales empleados y el presupuesto del mismo.
- **Capítulo 9:** Conclusiones. Se resumen en este capítulo las conclusiones obtenidas en este proyecto, los resultados alcanzados y los trabajos futuros.
- **Capítulo 10:** Acrónimos y Glosario. Definición de términos empleados y acrónimos.
- **Capítulo 11:** Bibliografía. Referencias consultadas en la elaboración del proyecto y la memoria.

Capítulo 2: Análisis de la Aplicación

2.1 Introducción

El objetivo final al que servirá este proyecto es desarrollar una aplicación para dispositivos móviles, que permitirá al usuario conocer la posición de sus contactos y mostrarlas sobre un mapa, en el instante que el usuario lo solicite. De la misma manera el usuario puede compartir su ubicación con los contactos que él decida.

En este capítulo se detallan las funcionalidades requeridas para dicha aplicación, los procesos que conllevan y la arquitectura del sistema completo.

2.2 Funcionalidades

Los requisitos de negocio o las principales funcionalidades que se desarrollan en la aplicación se definen en un formato adaptado al proyecto, acorde con la metodología definida por el IIBA [2]. Los requisitos de negocio se expresan a un nivel de abstracción mayor que los requisitos funcionales o no funcionales, describen las actividades y lógica de negocio sin detallar cuestiones de la implementación.

ID	RN-01
Nombre	Registro
Descripción	El usuario debe registrarse en la aplicación la primera vez que accede. Los datos para el registro serán el teléfono y la cuenta asociada al dispositivo (correo electrónico). De esta forma se podrá identificar al usuario dentro del sistema.

Requisito 1: RN-01

ID	RN-02
Nombre	Activar/Desactivar
Descripción	El usuario puede no tener la aplicación visible, pero seguirá funcionando, permitiendo que los resultados de las posiciones se obtengan en tiempo real.

	Este funcionamiento se puede activar o desactivar en cualquier momento por parte del usuario.

Requisito 2: RN-02

ID	RN-03
Nombre	Contactos
Descripción	El usuario debe conocer cuáles de entre sus contactos son usuarios de la aplicación, es decir, aquellos que ya están registrados.

Requisito 3: RN-03

ID	RN-04
Nombre	Privacidad
Descripción	El usuario puede decidir en todo momento a cuales de sus contactos permite conocer su posición y a cuales no.

Requisito 4: RN-04

ID	RN-05
Nombre	Mapa
Descripción	La posición del usuario y de los contactos que permiten al usuario conocer la suya, se mostrarán en un mapa. El mapa no tiene porqué ser adaptado a la realidad, puede ser temático, turístico, etc. El usuario podrá descargarse diferentes mapas desde la propia aplicación, dependiendo del contexto en que se encuentre.

Requisito 5: RN-05

ID	RN-06
Nombre	Actualización
Descripción	Cada vez que se ejecute la aplicación, se actualizará la información de los contactos, nuevos contactos y la información de los APs (puntos de acceso).

Requisito 6: RN-06

ID	RN-07
Nombre	Información
Descripción	La aplicación mostrará también información estática relacionada con la temática de la aplicación y del servicio: estado del servicio, información externa, etc.

Requisito 7: RN-07

2.3 Arquitectura

El proyecto engloba el desarrollo de un sistema completo, no solo la aplicación móvil. Es necesario introducir más elementos en el sistema para poder implementar todas las funcionalidades requeridas. La arquitectura del sistema es distribuida, de tipo cliente-servidor, es decir, parte de la lógica de la aplicación se ejecuta en un servidor externo. La comunicación es bidireccional y tanto el servidor como el dispositivo móvil pueden iniciar la comunicación.

A su vez la parte del servidor se descompone en varios subsistemas: uno que se encarga de ejecutar la lógica y otro que se encarga de la comunicación con los dispositivos móviles a través de notificaciones.

El sistema de notificaciones permitirá una comunicación más rápida y directa con el cliente, aumentando la eficiencia y velocidad del sistema.

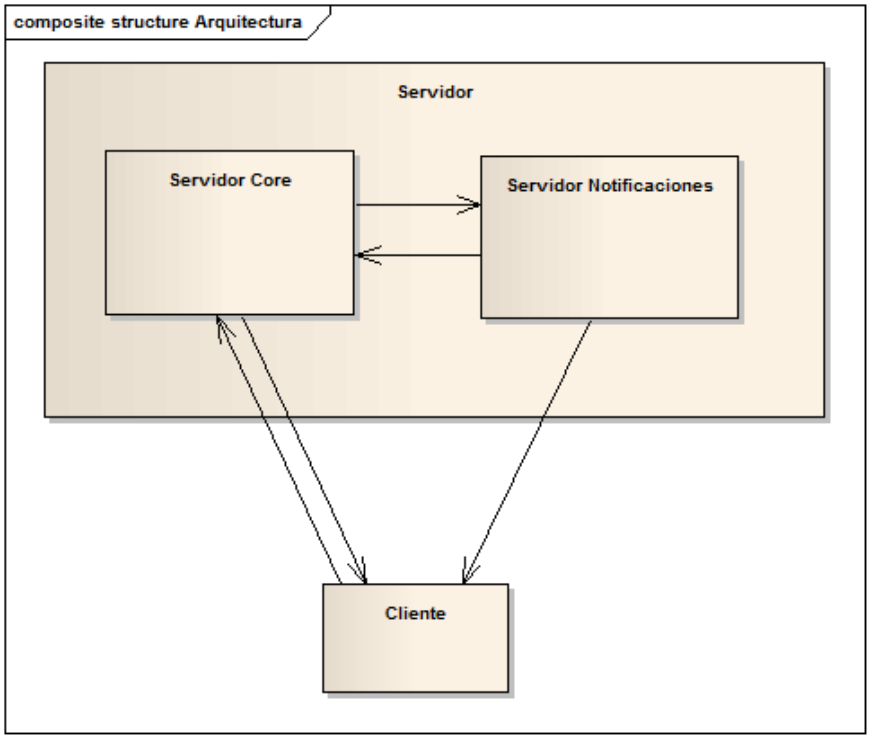

Figura 2: Arquitectura del sistema completo

2.3.1 Cliente

Es la aplicación que se ejecuta en el terminal móvil, a través de la cual interactúa el usuario. Recoge las peticiones del usuario y muestra la información de las posiciones en un mapa. Se comunica con el servidor *Core* para registrarse, actualizar información o permisos de los

usuarios, descargarse mapas, actualizar la información de los APs y para enviar la posición actual.

2.3.2 Servidor

Dentro de la parte que engloba al servidor se diferencian varios subsistemas, que realizan tareas suficientemente diferenciadas.

2.3.2.1 Subsistema *Core*

Ejecuta la lógica de la aplicación. Almacena la información de los usuarios, la privacidad, mapas, etc. Se comunica con el cliente para enviarle información de APs, mapas o nuevos contactos.

El servidor *Core* recibe las peticiones de localización de los contactos de un cliente. Cuando la recibe, comprueba a qué usuarios debe solicitar dicha información y se la envía a través del servidor de notificaciones. También las solicitudes de registro se envían al servidor de notificaciones.

2.3.2.2 Subsistema Notificaciones

Este subsistema se encarga de enviar notificaciones *push* a los dispositivos móviles que estén registrados en el servicio. Se enviarán notificaciones a los clientes para solicitar que envíen su posición actual o para informarles que se ha actualizado la posición de alguno de sus contactos.

2.4 Procesos

En este apartado se describen los procesos que se ejecutan en el sistema para soportar las funciones requeridas. Los procesos se detallan a alto nivel, reflejando los actores que intervienen y el intercambio de datos entre ellos.

2.4.1 Actores

En este subapartado se reflejan los actores que intervienen en los procesos y se define un identificador para cada uno de ellos.

ID	Nombre
USUARIO	Usuario de la aplicación
CORE	Servidor de aplicaciones en el que se ejecuta la lógica.
NOTIF	Servidor de notificaciones *push*.

Tabla 1: Actores

2.4.2 Procesos

Los procesos se relacionan con los requisitos, describen las actividades realizadas para cumplir los objetivos definidos en los requisitos de negocio.

ID	Identificador único del proceso
Nombre	Nombre del proceso
Descripción	Descripción del proceso
Requisito	Requisito asociado
Detonante	Evento que desencadena el proceso.
Actores	Actores involucrados en el proceso
Ejecución	Secuencia de acciones que se desarrollan en el proceso.

Proceso 1: Campos Proceso

ID	PR-01
Nombre	Registrar usuario
Descripción	Permite al usuario registrarse en el servicio de localización.
Requisito	RN-01, RN-03
Detonante	El usuario ejecuta la aplicación por primera vez.
Actores	USUARIO, CORE, NOTIF
Ejecución	El proceso se puede ver en la Figura 3. 1. El USUARIO lanza la aplicación 2. Se le solicita que introduzca los datos para la cuenta: correo, teléfono,.. 3. Los datos se envían al servidor CORE que los guarda 4. El servidor CORE renvía los datos al servidor NOTIF para registrarle en el servicio de notificaciones. 5. El servidor CORE confirma el alta al USUARIO 6. El cliente envía al servidor CORE una lista con los contactos de la agenda del USUARIO, para comprobar cuáles están dados de alta en el servicio. 7. Recibe como respuesta los usuarios registrados.

Proceso 2: PR-01

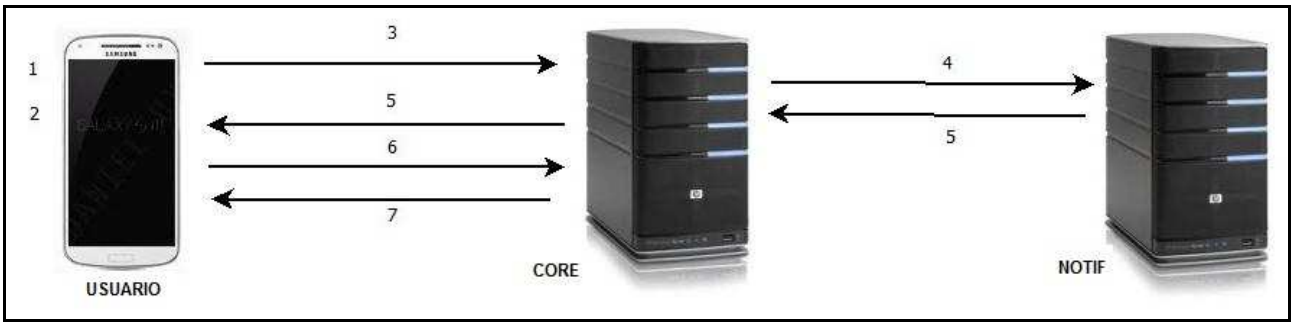

Figura 3: Proceso PR-01

ID	PR-02
Nombre	Actualizar información
Descripción	Actualiza la información de la aplicación relativa a los APs, a los contactos y otras informaciones.
Requisito	RN-01, RN-06, RN-07
Detonante	El USUARIO arranca la aplicación.
Actores	USUARIO, CORE
Ejecución	El proceso se puede ver en la Figura 4. 1. El USUARIO lanza la aplicación 2. Se envía al servidor CORE los nuevos contactos 3. El servidor CORE devuelve los usuarios registrados 4. El cliente envía al servidor CORE la versión del fichero de APs. 5. El servidor CORE devuelve el nuevo fichero de APs Alternativa: 5. En caso de no haber nueva versión devuelve una respuesta nula.

Proceso 3: PR-02

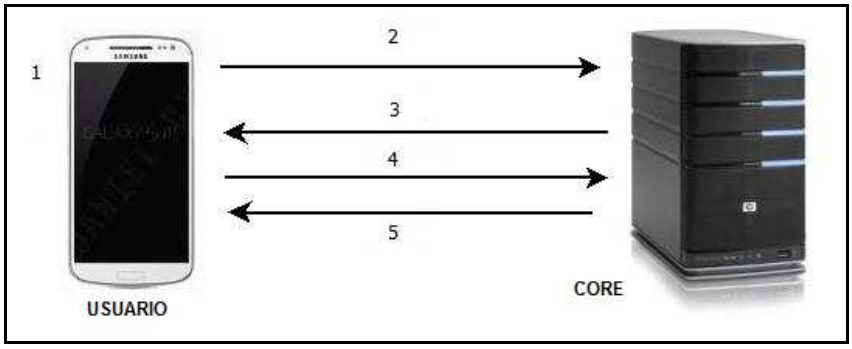

Figura 4: Proceso PR-02

ID	PR-03
Nombre	Definir privacidad contactos
Descripción	Guardar información sobre la privacidad de la información relativa a cada contacto.
Requisito	RN-04
Detonante	Configuración por parte del USUARIO.
Actores	USUARIO, CORE
Ejecución	1. El USUARIO selecciona la opción de privacidad relativa a otro usuario 2. Se envía al servidor CORE la información del cambio en la privacidad 3. El servidor CORE actualiza la información 4. El servidor CORE devuelve una respuesta como fin del proceso.

Proceso 4: PR-03

ID	PR-04
Nombre	Descargar mapas
Descripción	Descarga del servidor nuevos mapas.
Requisito	RN-05
Detonante	El USUARIO intenta obtener las posiciones de los contactos, pero no tiene el mapa necesario.
Actores	USUARIO, CORE
Ejecución	1. El USUARIO selecciona la opción de ver las posiciones de sus contactos 2. La aplicación detecta que no tiene el mapa sobre el que mostrar la posición

	actual del USUARIO. 2. Se envía al servidor CORE la petición del nuevo mapa 3. El servidor CORE devuelve el mapa al USUARIO 4. El USUARIO lo almacena y lo muestra.

Proceso 5: PR-04

ID	PR-05
Nombre	Obtener posición contactos
Descripción	Se obtiene la posición de los contactos con permisos suficientes y se muestran sobre el mapa apropiado.
Requisitos	RN-02, RN-05
Detonante	El USUARIO intenta obtener las posiciones de los contactos.
Actores	USUARIO, CORE, NOTIF
Ejecución	El proceso se puede ver en la Figura 5. 1. El USUARIO selecciona la opción de ver las posiciones de sus contactos 2. Se envía al servidor CORE la petición de las posiciones de los contactos. 3. El servidor CORE selecciona de entre los contactos del USUARIO, aquellos de los que tiene permiso para consultar la posición. 4. El servidor CORE envía la información de dichos usuarios al servidor NOTIF. 5. El servidor NOTIF envía a cada contacto recibido una notificación para que remita su posición actualizada. 6. Cuando cada usuario recibe la petición, obtiene la posición actualizada y se la envía al servidor CORE. 7. El servidor envía al USUARIO las nuevas posiciones. 8. La aplicación cliente va mostrando al USUARIO las nuevas posiciones en el mapa. Alternativas: 2. La aplicación detecta que no tiene el mapa sobre el que mostrar la posición actual del USUARIO. Se lanza el proceso PR-04.

Proceso 6: PR-05

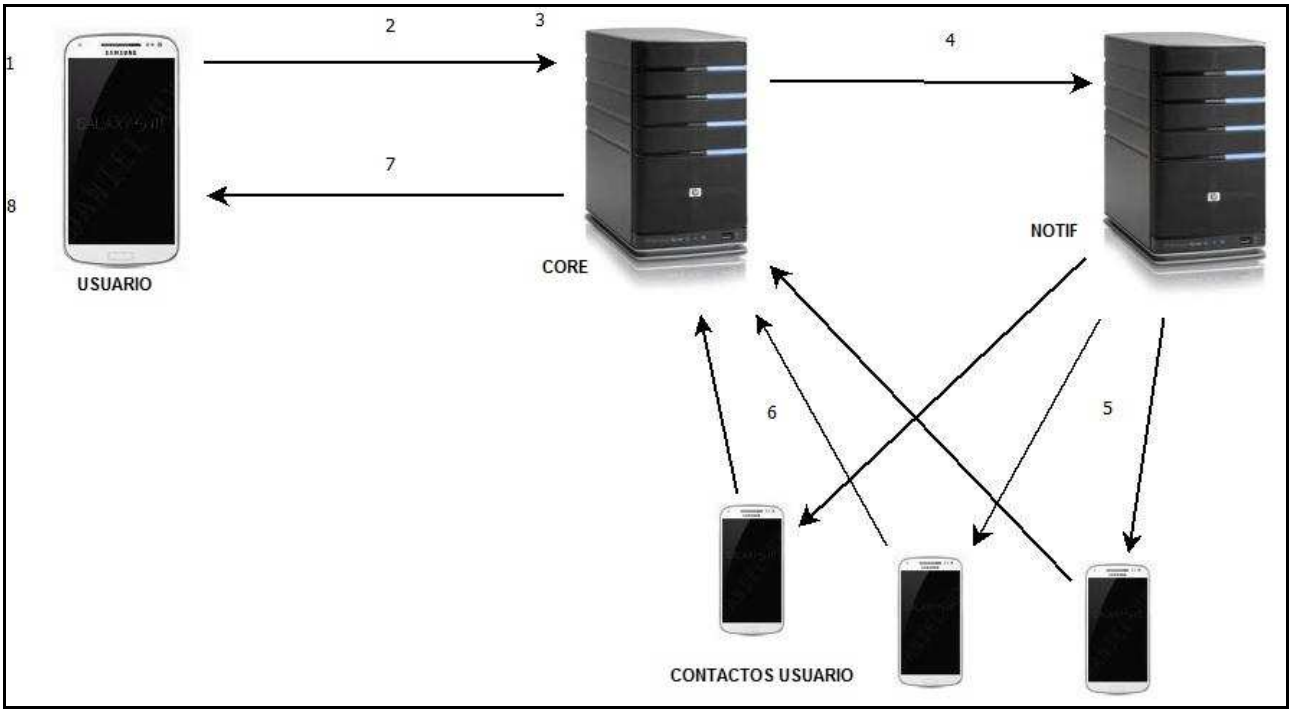

Figura 5: Proceso PR-05

Capítulo 3: Análisis Tecnológico

En este capítulo se va a analizar las diferentes tecnologías y productos que permitirán desarrollar el proyecto y construir cada uno de los módulos que lo componen.

3.1 Cliente

El cliente del sistema es probablemente la parte más importante de la aplicación, la elección de la tecnología sobre la que se desarrollará la parte cliente condicionará en gran medida el resto de tecnologías a utilizar en el resto de módulos.

Por su mayor popularidad, se consideran dos principales opciones como plataformas de desarrollo de aplicaciones móviles: *iOS* y *Android*.

3.1.1 iOS

iOS [3] es un sistema operativo móvil desarrollado originalmente para el iPhone por Apple, siendo después usado en otros dispositivos de la marca, ya que no se permite instalarlo en dispositivos de terceros.

Destaca por su estabilidad y por su interfaz de usuario muy rápida y fluida, con tiempos de respuesta muy bajos.

iOS se deriva de Mac OS X, un sistema operativo basado en Unix y cuenta con cuatro capas de abstracción: el núcleo del sistema operativo, la capa de "Servicios Principales", la capa de "Medios" y la capa de "Cocoa Touch".

Apple tiene un kit de desarrollo o SDK disponible permitiendo así a los desarrolladores hacer aplicaciones para el iPhone y demás dispositivos que utilizan este sistema operativo. El entorno de desarrollo incluye compiladores y un IDE específico (Figura 6) para desarrollar aplicaciones para productos de la firma, denominado Xcode.

Figura 6: Xcode[2]

Las aplicaciones se desarrollan en un lenguaje de programación llamado Objective-C, es un lenguaje orientado a objetos basado en el lenguaje C. Objective-C consiste en una capa muy fina situada por encima de C, por lo que es posible compilar cualquier programa escrito en C con un compilador de Objective-C, y también puede incluir libremente código en C dentro de una clase de Objective-C.

El kit de desarrollo incluye un simulador del iPhone que puede verse en la Figura 7, que permite probar las aplicaciones desarrolladas, sin embargo, para poder probar las aplicaciones en un dispositivo físico, es necesario registrarse como desarrollador y pagar la cuota del *iPhone Developer Program* de 99$ anuales. Y si se desea vender las aplicaciones a través de la tienda de Apple App Store, el 70% del precio de la aplicación se destina al desarrollador y el 30% para Apple.

[2] https://developer.apple.com/xcode/ (Junio 2012)

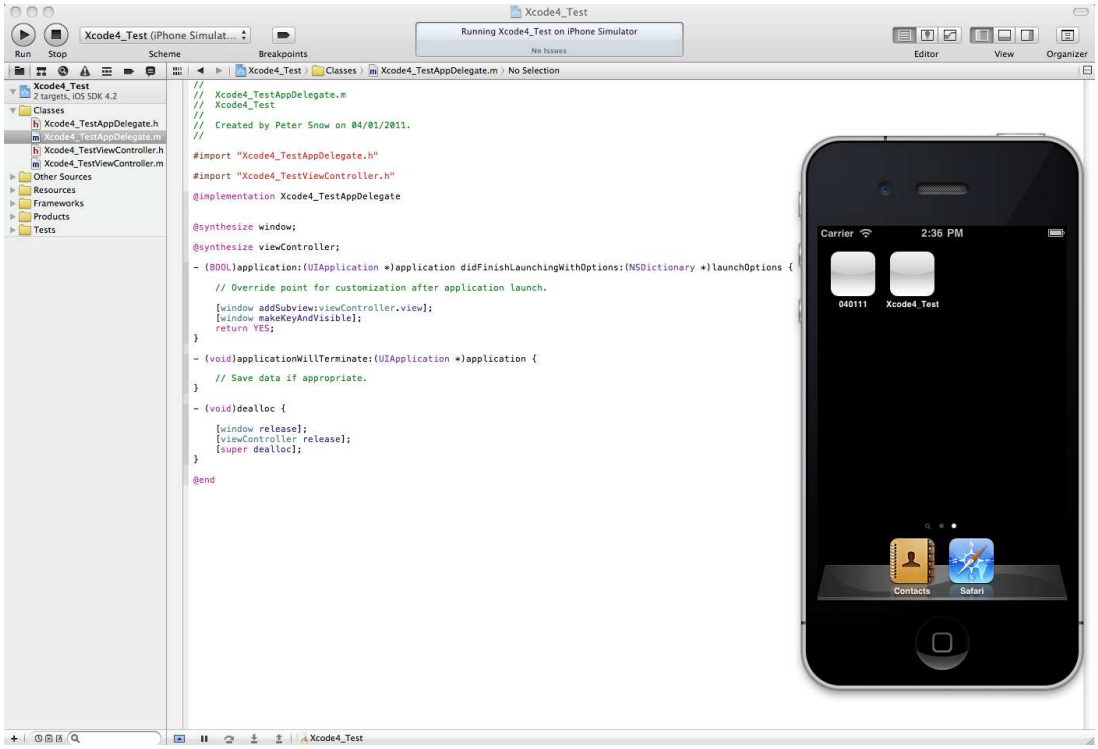

Figura 7: Simulador iPhone

El kit de desarrollo está únicamente disponible para instalarse en ordenadores con el sistema operativo MAC OS X, también propio de Apple.

3.1.2 Android

La plataforma *Android* [4] es el producto del trabajo de un grupo de organizaciones, liderado por Google, que incluye a operadores móviles, fabricantes de móviles, fabricantes de componentes, empresas de software y otros proveedores. Este grupo de empresas colaboran para fabricar mejores teléfonos móviles y desde el punto de vista de desarrollo de software, *Android* se encuentra a la vanguardia del desarrollo de código abierto.

Android no es solamente un sistema operativo para dispositivos móviles, también incluye una pila de software, middleware y un conjunto de aplicaciones muy destacadas. El SDK de *Android* proporciona las herramientas y APIs necesarias para empezar a desarrollar aplicaciones en la plataforma *Android* usando el lenguaje de programación Java.

Principales características:

- Arquitectura de aplicaciones que permite la reutilización y sustitución de componentes
- Máquina virtual Dalvik optimizada para dispositivos móviles
- Navegador integrado basado en el motor WebKit de código abierto
- Gráficos optimizados 2D, gráficos 3D basado en OpenGL

- SQLite para almacenamiento de datos
- Soporte para formatos comunes de audio, videos e imagen
- Telefonía
- Bluetooth, EDGE, 3G y WiFi
- Cámara, GPS, brújula y acelerómetro
- Entorno de desarrollo completo que incluye un emulador de dispositivos (Figura 8), herramientas para la depuración, la memoria y perfiles de rendimiento y un *plugin* para el IDE de Eclipse.

Figura 8: Emulador *Android*[3]

Además cuenta con una fuerte integración de todos los servicios de Google, como Gmail, Maps, Calendar, etc. La Figura 9 muestra una vista simplificada de la arquitectura del sistema de *Android*.

[3] http://developer.android.com/images/emulator-wvga800l.png (Junio 2012)

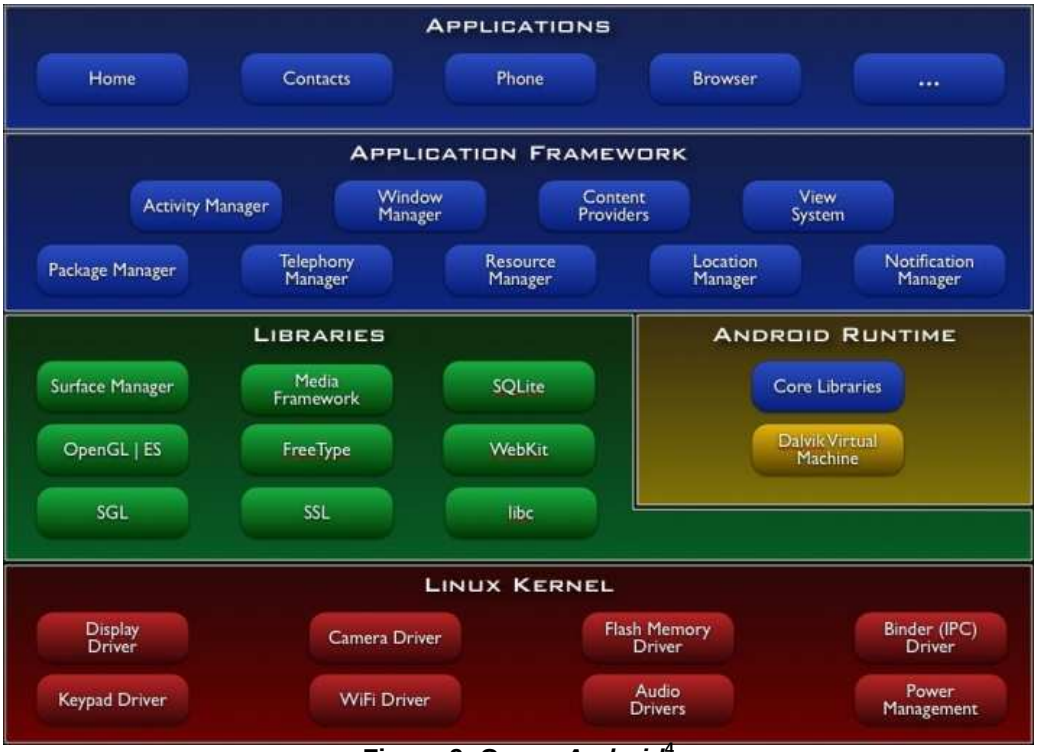

Figura 9: Capas *Android*[4]

Para publicar aplicaciones en el sistema de venta de aplicaciones de *Android*, llamado Google Play, es necesario darse de alta como desarrollador, pero no requiere el pago de ninguna cuota. Google aplica una tarifa de transacción del 30% del valor de la aplicación.

3.1.3 Mercado

Android mantiene su posición de liderazgo como el más popular de los sistemas operativos para *smartphone* en España. Según las estadísticas de ventas de terminales publicadas [5], la cuota de *Android* en el mercado mundial durante los tres primeros meses de este año se sitúa ligeramente por encima del 50%, cuando hace un año era del 44,6%.

El estudio indica que el principal baluarte de *Android* se encuentra en Europa, y muy especialmente en España, donde ostenta el 72,3% del mercado gracias a un crecimiento interanual del 39,5%, como se observa en el gráfico siguiente:

[4] http://developer.android.com/images/system-architecture.jpg (Junio 2012)

Figura 10: Venta de *Smartphones* en España[5]

3.1.4 Elección: Android

Uno de los principales motivos para elegir el *Android* como plataforma sobre la que desarrollar la parte cliente del sistema es el mayor número de usuarios potenciales, dada su posición dentro del mercado.

Pero no es la única razón, *Android* es una plataforma open source, por tanto, no requiere inversión en licencias o equipamiento extra, al contrario que iOS que requiere de licencia de desarrollador para instalar la aplicación en los dispositivos móviles.

Así mismo, podemos trabajar con el entorno de desarrollo sobre cualquier plataforma, sin necesidad de inversión en equipamiento específico. No requiere instalarse sobre un entorno propietario como es el caso de MAC OS X.

Por el hecho de ser *open source*, existen un gran volumen de documentación y recursos técnicos de libre disposición y una comunidad de desarrolladores con experiencia, lo que facilitará la solución de problemas en la fase de desarrollo.

La última de las razones para su elección proviene de una preferencia personal del autor hacia este sistema operativo.

[5] http://www.eleconomista.es/CanalPDA/files/smartphones_espan%CC%83a_kantar.jpg (Junio 2012)

Además, la elección de *Android* nos facilita la integración con otros servicios de Google, que podrán ser de utilidad en otros módulos de la aplicación.

La elección de *Android* como sistema sobre el que desarrollar la aplicación, condiciona en parte la elección de las tecnologías del resto de módulos.

3.2 Servidor

Para la parte del servidor existen múltiples opciones, desde servidores web básicos a completos servidores de aplicaciones web, con soporte para distintos lenguajes más allá de HTML, como servidores para PHP, Python, Ruby, servidores que soporten Java y JSP.

3.2.1 Apache

El servidor HTTP Apache [6] es un servidor web HTTP gratuito y de código abierto, que está disponible disponible para múltiples plataformas (Windows, Unix, Macintosh, etc.).

Apache presenta entre otras características su alta capacidad de configuración, su diseño modular, bases de datos de autenticación y negocio, lo que le proporciona una amplia aceptación en la red, siendo uno de los servidores HTTP más usados.

Algunas de sus principales ventajas son:

- Modular
- Código abierto
- Multi-plataforma
- Muy extendido, gran cantidad de información y soporte

3.2.2 Tomcat

El servidor Tomcat [7] es un servidor web gratuito y de código libre que incluye soporte para servlets y JSPs, no se trata de un servidor de aplicaciones. Es multiplataforma, puede ejecutarse sobre cualquier sistema operativo, instalando previamente una máquina virtual de Java.

Tomcat es el servidor Web más utilizado a la hora de trabajar con Java en entornos web y además de JSPs y servlets, también permite ejecutar servicios web con Axis.

Las principales características que presenta son:

- Autenticación de acceso básico.
- Negociación de credenciales.
- HTTPS
- Alojamiento compartido.
- CGI o interfaz de entrada común.
- Servlets de Java.

3.2.3 Google App Engine

Google App Engine [8] es un entorno de ejecución de aplicaciones web que usa las infraestructuras de Google y que proporciona a los desarrolladores una serie de ventajas importantes sobre otros servidores de aplicaciones.

Permite desarrollar y ejecutar ágilmente aplicaciones, incluso con pesadas cargas de trabajo y grandes cantidades de datos. App Engine incluye las siguientes funciones:

- Servidor web dinámico, totalmente compatible con las tecnologías web más comunes
- Almacenamiento permanente con funciones de consulta, clasificación y transacciones
- Escalado automático y distribución de carga
- API para autenticación de usuarios
- Entorno de desarrollo local que simula Google App Engine en local
- Tareas programadas para activar eventos en momentos determinados y a intervalos regulares.

Las aplicaciones se pueden ejecutar sobre un entorno Java que proporciona protocolos estándar, tecnologías comunes para el desarrollo de aplicaciones web y otras herramientas:

- Un SDK Java para App Engine y los componentes relacionados.
- Un plugin para Eclipse que permite desarrollar y probar aplicaciones para Google App.
- Aplicaciones de prueba
- Soporte para Servlets y JSPs.

3.2.2 Elección: Google App Engine

A pesar de existir gran cantidad de servidores web y de aplicaciones, Google App Engine proporciona una serie de características que lo diferencian del resto y que lo hacen muy atractivo a la hora de buscar un servidor donde desplegar una aplicación. Destaca su facilidad para iniciar el desarrollo, con aplicaciones de ejemplo y plugin para Eclipse, no requiere grandes tareas de administración ni configuración ni de creación de bases de datos. Dispone de escalabilidad automática, por lo que permite aprovechar las mismas tecnologías escalables sobre las que están creadas las aplicaciones de Google, con las mismas características de fiabilidad y rendimiento. Además es gratuito mientras no se superen unas cuotas de uso (cinco millones de vistas mensuales) alejadas de las expectativas de uso de esta aplicación.

3.3 Datos

Para la capa de datos, el servidor Google App Engine permite varias posibilidades, en principio muy parecidas, que se describen a continuación.

3.3.1 JDO

La interfaz de objetos de datos de Java (JDO) es una interfaz estándar para almacenar objetos en una base de datos. El estándar permite configurar las interfaces a través de anotaciones en los objetos Java, recuperar de objetos a través de consultas y permite interactuar con la base de datos mediante transacciones.

Una aplicación que utilice la interfaz de JDO puede funcionar con distintos tipos de bases de datos sin usar ningún código específico para estas, incluidas las bases de datos relacionales, jerárquicas y de objetos.

3.3.2 JPA

Java Persistence API (JPA) es la API de persistencia desarrollada para la plataforma Java EE, es un framework que maneja datos relacionales en aplicaciones Java.

El objetivo que persigue el diseño de esta API es no perder las ventajas de la orientación a objetos al interactuar con una base de datos, siguiendo el patrón de mapeo objeto-relacional y permitiendo usar objetos regulares, conocidos como POJOs.

3.3.3 Elección: JDO

La gran ventaja de este tipo de interfaces, es la abstracción del tipo de base de datos, simplificando la migración de una aplicación entre diferentes soluciones de almacenamiento, en caso de posteriores modificaciones.

Trabajando en el entorno de Google App Engine, se plantean dos posibles soluciones para el almacenamiento de datos: JDO y JPA. Son soluciones que aportan similares características, posiblemente JDO un poco más completa y sea una tecnología un poco más madura, por su mayor trayectoria. Pero el motivo por el que se decide emplear JDO es por la mayor cantidad y detalle de documentación aportada por el propio servidor de aplicaciones, así como ejemplos en código, que facilitarán en gran medida el desarrollo y la integración en esta plataforma.

3.4 Servidor de notificaciones

Para el servidor de notificaciones se podría emplear un servidor web o de aplicaciones igual que para el servidor CORE y desarrollar una aplicación que realizara estas funciones. Pero existe un servicio específico de Google que implementan estas funcionalidades y que está orientado a notificaciones a dispositivos móviles *Android*: C2DM.

3.4.1 C2DM

Android Cloud To Device Messaging (C2DM) [9] es un servicio que permite a los desarrolladores enviar datos desde los servidores de aplicaciones a los dispositivos *Android*. El

servicio proporciona un mecanismo sencillo y ligero que los servidores pueden usar para decirle a las aplicaciones móviles que se pongan en contacto con el servidor directamente, en busca de actualizaciones o datos. El servicio de C2DM maneja todos los aspectos de la formación de colas de mensajes y la entrega a la aplicación de destino que se ejecuta en el dispositivo.

C2DM permite a terceros enviar mensajes ligeros a sus aplicaciones para *Android*. Se trata de un servicio de mensajería *push* que aprovecha la conexión de Google existente en los dispositivos. El servicio no está diseñado para enviar una gran cantidad de contenido a través de los mensajes. Por el contrario, se debe utilizar para indicar a la aplicación que hay nuevos datos en el servidor, de modo que la aplicación pueda tomar las medidas oportunas.

C2DM no ofrece ninguna garantía sobre la entrega o el orden de los mensajes. Por ejemplo en una aplicación de mensajería se usaría para indicar a la aplicación que el usuario tiene, mensajes nuevos, pero no enviaría el mensaje en sí, por su falta de confirmación en la entrega.

Una aplicación *Android* no necesita estar en ejecución para recibir los mensajes C2DM. El sistema activará la aplicación a través de una llamada cuando el mensaje llegue, siempre y cuando la aplicación está configurada con el receptor adecuado y permisos.

Limitaciones:

- Se requiere que los dispositivos tengan *Android* 2.2 o superior.
- C2DM utiliza la conexión existente de Google, por lo que es necesario estar autenticado con una cuenta Google para poder utilizarlo. Es el mismo servicio de mensajes push que utilizan aplicaciones como GMail o Google Maps.
- El tamaño máximo de los mensajes son 1024 bytes.
- El número de mensajes es limitado, Google establece unas cuotas mensuales

3.4.2 Elección: C2DM

No hay alternativas para la funcionalidad buscada, la alternativa sería una aplicación propia desarrollada a para este propósito. Pero este servicio de Google cumple a la perfección como servicio de envío de notificaciones para garantizar ciertas funcionalidades requeridas en la aplicación. Además de estar diseñado para dispositivos *Android* y perfectamente integrado con Google App Engine, es sencillo de usar.

Como desventajas, C2DM tiene ciertas limitaciones, pero son perfectamente asumibles: la versión de *Android* requerida no es nueva (se han liberado cuatro versiones posteriores), para usar un dispositivo *Android* se necesita una cuenta de Google, el tamaño de los mensajes no es inconveniente ya que son solo notificaciones y las cuotas de mensajes que establece Google son suficientes para el alcance de este proyecto (200000 mensajes diarios).

3.5 Lenguaje de programación

El lenguaje con el que se desarrollarán las aplicaciones.

3.5.1 Python

El lenguaje Python fue creado a finales de los ochenta por Guido van Rossum en el Centro para las Matemáticas y la Informática (CWI, Centrum Wiskunde & Informatica). El nombre proviene de la afición de su creador por los humoristas británicos Monty Python.

Python es un lenguaje de programación similar a Perl, pero con una sintaxis más limpia y que favorece un código legible. Se trata de un lenguaje interpretado o de script, con tipado dinámico, fuertemente tipado, multiplataforma y orientado a objetos.

Python es un lenguaje muy potente, con gran cantidad de librerías disponibles que hacen que desarrollar una aplicación sea sencillo y muy rápido.

3.5.2 JAVA

Java es un lenguaje de programación orientado a objetos, desarrollado por Sun Microsystems a principios de los años 90. El lenguaje en sí mismo toma mucha de su sintaxis de C y C++, pero tiene un modelo de objetos más simple y elimina herramientas de bajo nivel, que suelen inducir a muchos errores, como la manipulación directa de punteros o memoria. La memoria es gestionada mediante un recolector de basura.

Java es independiente de la plataforma, los programas escritos en este lenguaje pueden ejecutarse igualmente en cualquier tipo de hardware. Para ello, se compila el código fuente escrito en lenguaje Java, para generar un código conocido como "bytecode", instrucciones máquina simplificadas específicas de la plataforma Java. El bytecode es ejecutado entonces en la máquina virtual (JVM), un programa escrito en código nativo de la plataforma destino, que interpreta y ejecuta el código.

La versatilidad y eficiencia de la tecnología Java, la portabilidad de su plataforma y la seguridad que aporta, la han convertido en la tecnología ideal para su aplicación a redes. Más de 4.500 millones dispositivos utilizan la tecnología Java.

Java ha sido probado, mejorado, ampliado y probado por una comunidad especializada de más de 9 millones de desarrolladores. Su uso es especialmente extendido en la parte servidor, pero se puede utilizar en todos los entornos.

3.5.3 Elección

En la parte cliente, dado que hemos elegido como plataforma *Android*, la elección del lenguaje es inmediata, ya que *Android* proporciona un API en Java par el desarrollo de aplicaciones.

Para el servidor existen más opciones, pero eligiendo como plataforma Google App Engine, se limitan a dos: Python o Java. La gran ventaja de elegir Java es que unificamos el lenguaje con el que se desarrollan todas las partes de la aplicación.

Por otra parte, cabe destacar que Java es un lenguaje con el que el desarrollador de este proyecto tiene una dilatada experiencia, lo que le convierte en la elección más clara.

3.6 IDE

El IDE es el entorno de desarrollo, es el programa que utilizarán los desarrolladores para programar las aplicaciones. Se compararán algunos entornos de desarrollo para Java.

3.6.1 Netbeans

Netbeans [10] es un entorno integrado gratuito y de código abierto, creado por Sun MicroSystems, para desarrollar aplicaciones software. Contiene las herramientas necesarias para crear aplicaciones tanto de escritorio, empresariales, web y aplicaciones móviles a nivel profesional. Tiene soporte para varios lenguajes de programación, entre ellos C / C++, PHP, JavaScript y Groovy, pero está diseñado especialmente para Java.

Netbeans es modular, cada módulo proporciona una función bien definida, tales como el soporte de Java, edición, o soporte para el sistema de control de versiones. También soporta paquetes adicionales que se pueden descargar individualmente y que permiten extender funcionalidades.

3.6.2 Eclipse

Eclipse [11] es un entorno de desarrollo integrado de código abierto y multiplataforma. Es una base o armazón (workbench) sobre el que se pueden montar herramientas de desarrollo para cualquier lenguaje, mediante la incorporación de los plugins.

La arquitectura de plugins además de integrar diversos lenguajes, permite introducir otras funcionalidades que pueden resultar útiles durante el proceso de desarrollo como: herramientas UML, editores visuales de interfaces, ayuda en línea para librerías, etc.

Eclipse fue creado originalmente por IBM y ahora es desarrollado por la Fundación Eclipse, una organización independiente sin ánimo de lucro que fomenta una comunidad de código abierto y un conjunto de productos y servicios que lo complementan.

Eclipse es también una comunidad de usuarios, dedicada a extender constantemente las áreas y funcionalidades que cubre la aplicación.

3.6.3 Elección

Los dos IDEs comparados están especialmente enfocados al lenguaje Java, pero la elección de Eclipse como entorno para desarrollar el proyecto fue muy clara, por lo experiencia que acumula el desarrollador del proyecto con esta herramienta. Pero sobre todo, por la existencia de plugins de Google que ayudan a desarrollar aplicaciones para los entornos y servicios de *Android* y App Engine, lo que facilita en gran medida la parte de programación.

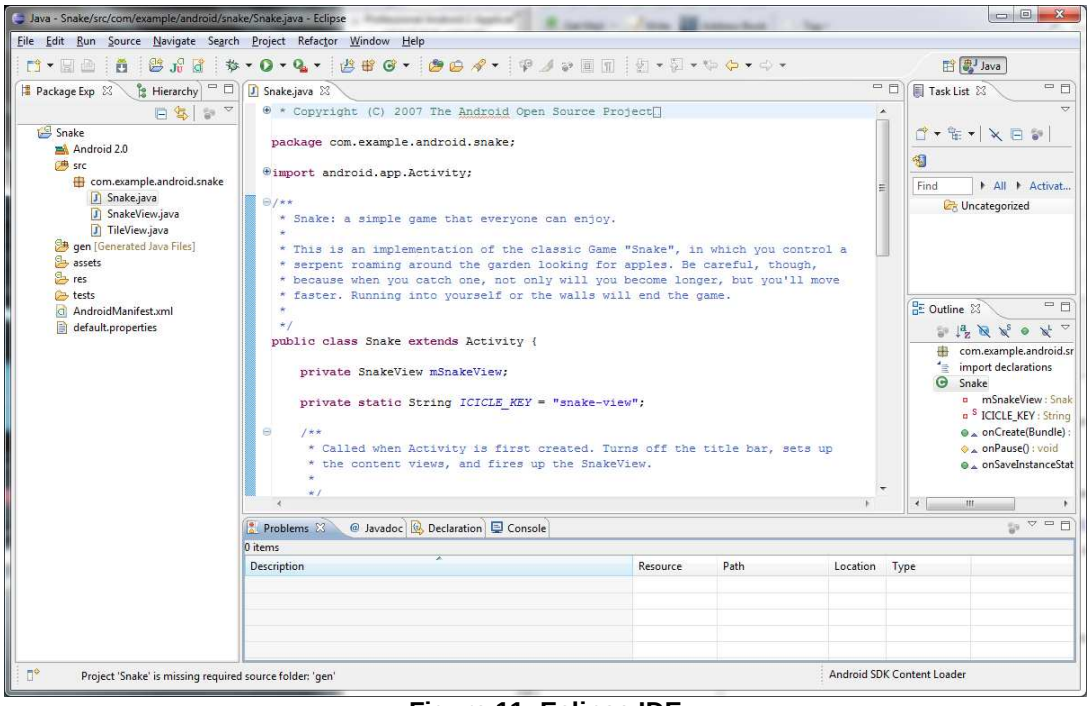

Figura 11: Eclipse IDE

3.7 Otras consideraciones

Para las tecnologías anteriores, se definen las versiones que se utilizarán en este proyecto. Además de las tecnologías, existen otras consideraciones a tener en cuenta para que el funcionamiento del sistema sea correcto: las características de los dispositivos móviles.

3.7.1 Versiones

Se detallan en la Tabla 2 la versión de cada una de las tecnologías utilizadas.

Tecnología	Versión
S.O. dispositivo móvil	Android versión 2.2 o superior
Servidor	Google App Engine - versión JVM 6
Almacenamiento de datos	JDO 2.3
Servidor de notificaciones	C2DM
Lenguaje de programación	Java 1.6
IDE	Eclipse Indigo (3.7)

Tabla 2: Versiones

3.7.2 Requisitos de los dispositivos móviles

Para el correcto funcionamiento de todas las funcionalidades de la aplicación, es necesario que los dispositivos móviles dispongan de una serie de características:

3.7.2.1 Android

Dado que la aplicación se ha desarrollado para esta plataforma y que el servicio que proporciona Google C2DM de notificaciones es solo para usuarios de *Android*, la aplicación se limita a los usuarios que dispongan de este sistema operativo.

3.7.2.2 GPS

Como se pretende obtener la posición, los usuarios deben tener GPS y además tenerlo encendido.

3.7.2.3 WIFI

Para poder hacer uso del sistema de posicionamiento en interiores es necesario que el dispositivo tenga una interfaz de red inalámbrica 802.11.

3.7.2.4 Conexión de datos

Para poder intercambiar información con el servidor, es necesario que dispongan de conexión a Internet.

Capítulo 4: Características Diferenciadoras

En este capítulo se analizarán las aplicaciones existentes actualmente con finalidades similares y se destacarán las características que diferencian a la aplicación que se va a desarrollar de sus competidoras.

4.1 Estado del arte

Android tiene disponibles a través de su plataforma de descarga de aplicaciones, Google Play, más de 450.000 aplicaciones [12]. De entre todo este conjunto de aplicaciones son numerosas las que están relacionadas con la localización y el posicionamiento del usuario, incluso Google tiene una aplicación con este fin, llamada Google Latitude.

Se verán a continuación algunas de las aplicaciones más extendidas y sus principales características. Estas aplicaciones están disponibles para la plataforma, no únicamente *Android*, veremos las características comunes de las aplicaciones en las diferentes plataformas.

4.1.1 Foursquare

Foursquare [13] es una aplicación gratuita que permite compartir y guardar los sitios que el usuario va visitando. Además le ofrece recomendaciones personalizadas y ofertas según la posición en la que se encuentra, sus preferencias y las de sus amigos.

Foursquare fué de las primeras aplicaciones que surgieron para localizar y compartir localizaciones entre usuarios. Actualmente cuenta con más de 20 millones de usuarios y está disponible para las siguientes plataformas: *iOS* (Figura 12), *Android*, *Blackberry, WebOS, Symbian, Meego* y *Windows Phone*.

Está principalmente orientada a descubrir y recomendar comercios entre los usuarios, su plataforma la usan más de 750.000 comerciantes. La clave de Foursquare es convertir el hecho de ir marcando todos los sitios visitados en un juego. Cada vez que se visita algún lugar: bar, discoteca, aeropuerto, hotel,… se obtiene cierta cantidad de puntos.

Figura 12: Foursquare[6]

4.1.2 Facebook Places

Facebook Places [14] es una funcionalidad que puede usarse tanto desde los *smartphones* como desde la propia web de Facebook y que permite a los usuarios de Facebook compartir su ubicación. Esta funcionalidad se integra dentro de la aplicación de Facebook y está disponible para todas las plataformas que disponen de aplicación de Facebook: *iOS* (Figura 13), *Android, Blackberry, WebOS, Symbian, Meego* y *Windows Phone*.

El servicio Facebook Places tiene tres objetivos básicos:

- Los usuarios pueden compartir su posición en un mapa con otros usuarios.
- Permite así saber cuáles de los amigos están cerca, donde y cuando.
- Ayudar a los usuarios a descubrir sitios nuevos a su alrededor.

Entre las posibilidades de configuración, permite activar o desactivar el servicio de ubicación y definir quién puede ver la información que compartes. La gran ventaja de esta aplicación frente a otros competidores, es la gran cantidad de usuarios que ya utilizan Facebook y que no necesitan registrarse en otro servicio ni descargarse otra aplicación.

[6] https://ss1.4sqi.net/img/homepage/2-friends-652c35df0a39a12dfa77ee0e6b47e59d.png (Junio 2012)

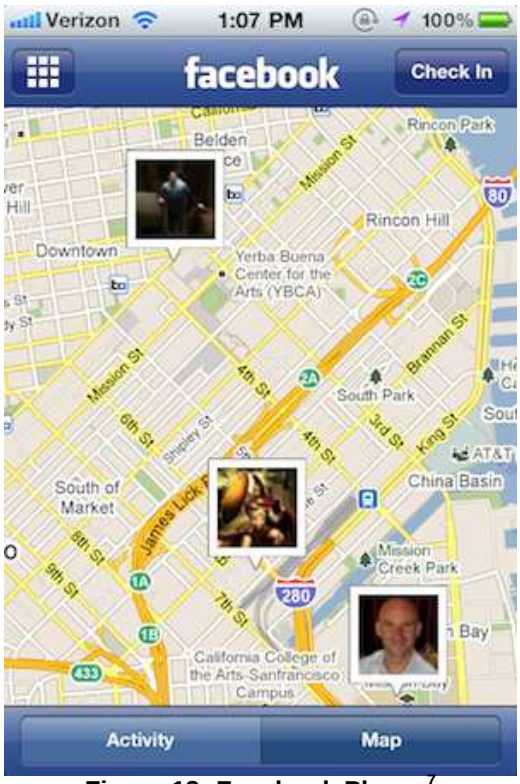

Figura 13: Facebook Places[7]

4.1.3 Google Latitude

Para su uso en dispositivos móviles, el servicio de Latitude [15] está integrado dentro de la aplicación Google Maps, la aplicación de mapas y navegación GPS de Google. Según la propia web de Google Maps, Latitude permite descubrir la posición de tus amigos ahora mismo y compartir tu propia ubicación o registrar tus visitas para que tus amigos sepan dónde estás.

Viendo más en detalle sus características, permite:

- Descubrir en el mapa la posición actual de tus amigos
- Compartir tu propia ubicación
- Configurar la privacidad: definir quién puede ver tu ubicación
- Controlar tu ubicación: ocultarla o compartirla en cualquier momento
- Visitar un sitio y compartir la ubicación
- Recibir notificaciones de sitios cercanos

[7] http://www.marketingactiva.com/web/wp-content/uploads/2011/04/facebook-places.png (Junio 2012)

Figura 14: Google Latitude[8]

4.2 Diferencias

Las aplicaciones vistas anteriormente proporcionan una serie de características y funcionalidades comunes, que también son básicas en la aplicación que se va a desarrollar.
Sin embargo, las aplicaciones anteriores están orientadas a compartir la ubicación en lugares de interés, recibir información de comercios cercanos o compartir información a través de redes sociales. Para los escenarios en que se pretende utilizar la aplicación a desarrollar, se añadirán unas nuevas características que la diferenciarán de las aplicaciones analizadas anteriormente.

4.2.1 Mapas personalizados

En la mayoría de las ocasiones los usuarios acceden a los servicios de geoposicionamiento para comprobar donde están sus amigos, buscar un restaurante cercano o compartir su posición actual. Para este tipo de situaciones los mapas físicos o urbanos, que muestran las calles de las ciudades, caminos o los lugares de interés, son apropiados.

Existen otros escenarios donde estos mapas no sirven o no contienen la información suficiente, lugares como estaciones de esquí, recintos feriales, parques de atracciones, festivales de

[8] http://www.google.com/intl/es_ALL/mobile/latitude/latitude-check-in-share.jpg (Junio 2012)

música y otros recintos cerrados. En estos lugares también es de gran utilidad conocer la posición de los amigos, pero se necesitan mapas de los recintos para situar a los usuarios en su posición real.

Una de las características que diferencia la aplicación que se va a desarrollar frente a las alternativas existentes, es el empleo de mapas personalizados. Se trata de mostrar la ubicación del usuario sobre un mapa que represente adecuadamente la distribución de espacios en estos entornos especiales.

La solución pasa por asociar a cada punto del mapa representativo del escenario en cuestión, una coordenada geográfica. De esta manera, podremos hacer corresponder las coordenadas recibidas por los sistemas de posicionamiento, como el GPS, con puntos del mapa personalizado.

4.2.1.1 Generación de mapas personalizados

Para obtener un mapa personalizado, que se corresponda con la situación geográfica real, partiremos de un mapa o plano esquemático del recinto, que puede no estar a escala ni guardar proporciones reales.

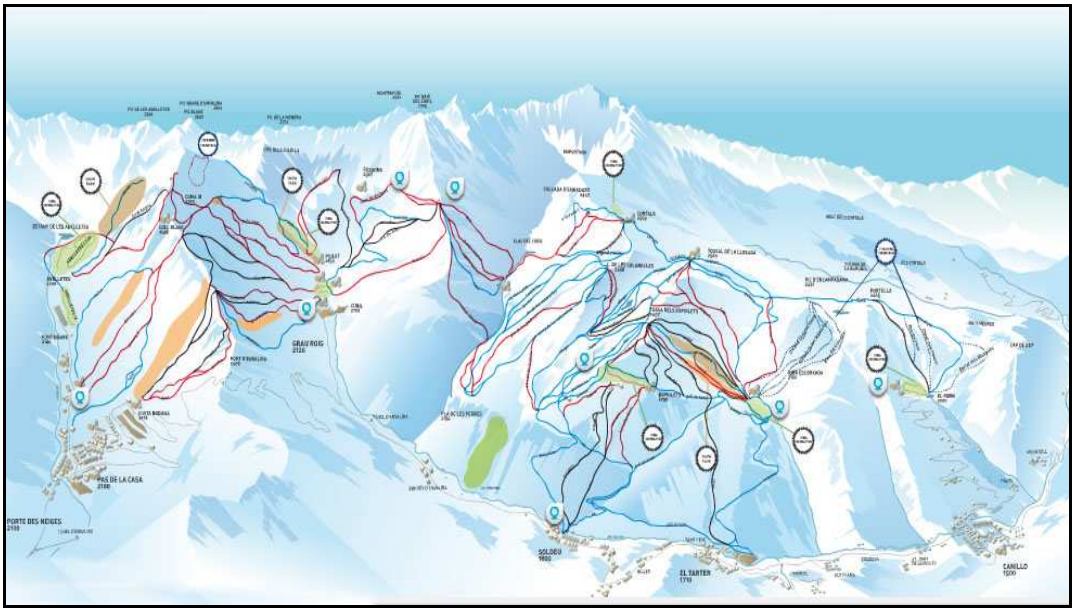

Figura 15: Mapa original

A partir de los puntos del mapa esquemático y de sus correspondientes puntos en un mapa geográfico, con coordenadas de longitud y latitud, se utilizan algoritmos de minería de datos que extrapolan iterativamente las correspondencias a todos los puntos del mapa.

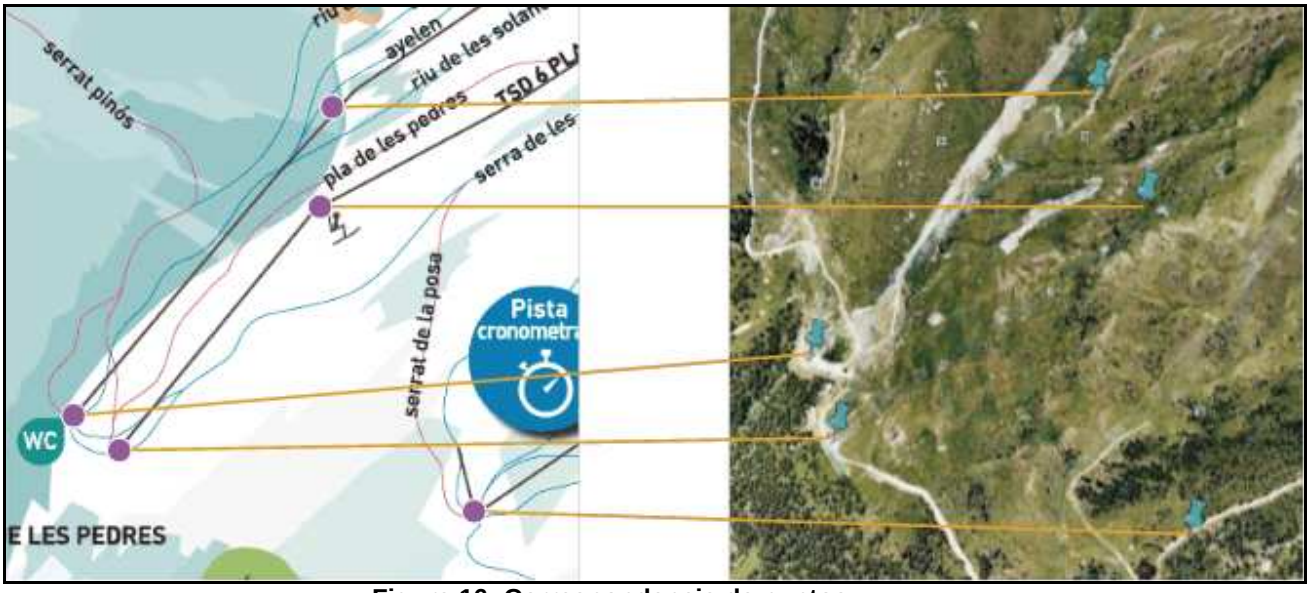

Figura 16: Correspondencia de puntos

A través de este sistema se podrán utilizar en la aplicación mapas personalizados que se correspondan con las coordenadas geográficas reales.

Toda la información referente a los algoritmos de interpolación utilizados y el procedimiento detallado se puede encontrar en el proyecto que se desarrolla conjuntamente con este[9].

4.2.2 Tiempo real

Otro de los aspectos en los que las aplicaciones existentes pueden mejorarse, es en la actualización de la posición de los usuarios. La forma y frecuencia de actualización se deja en manos de los usuarios y cuando un usuario quiere conocer la información de sus contactos, lo hace de forma pasiva, como lector de la información disponible, que puede ser de momentos muy anteriores al actual.

Sería de mayor utilidad si se pudiera conocer de forma inmediata la posición de los contactos cuando un usuario lo solicite, de forma activa. A este fin, el de poder disponer de la información actualizada, en tiempo real, sirve el servidor de notificaciones.

A partir de las últimas actualizaciones, Latitude permite la actualización en tiempo real, pero de forma pasiva, es decir, cada usuario va continuamente actualizando su información, con los consiguientes gastos de batería y datos que eso supone. Además no es seguro que todos los contactos lo vayan a hacer así.

Como ya se ha mencionado, se plantea una opción diferente: obtener los datos de las posiciones de los contactos bajo demanda, sólo en el instante que el usuario lo solicite y sin requerir la interacción de sus contactos.

[9] PFC – Francesco Gatti Gómez

La clave para conseguir este propósito pasa por no usar un mecanismo de *polling*, sino un servicio de *push* de mensajes. Se utiliza el servicio C2DM proporcionado por Google, que permite enviar mensajes desde un servidor a las aplicaciones en dispositivos *Android*.

Al no necesitar un continuo envío de datos, se limita el envío de mensajes en un instante concreto, tanto el consumo de batería como el de datos es muy bajo.

4.2.2.1 Funcionamiento básico

El funcionamiento básico que se ha planteado es un funcionamiento bajo demanda. Cuando un usuario desea conocer la posición de sus contactos, lanza una petición al servidor. El servidor comprueba la configuración de privacidad y envía una solicitud de la posición a los contactos permitidos. Cuando un contacto recibe la petición, sin necesidad de interactuación del usuario, el dispositivo móvil obtiene su posición GPS y la devuelve al servidor. Según el servidor va recibiendo las distintas posiciones de los contactos, las va remitiendo al usuario que las solicitó y la aplicación las va dibujando sobre el mapa.

4.2.2.2 Funcionamiento alternativo

El funcionamiento básico antes descrito puede requerir cierto tiempo de respuesta hasta que se obtienen todas las posiciones GPS. Se plantea un modo de funcionamiento alternativo, que obtiene resultados en tiempo semi-real. El funcionamiento consiste en que los usuarios actualicen su posición a intervalos de tiempo suficientemente cortos como para que su posición sea muy cercana a la actual, por ejemplo cada 1-2 minutos.

Se perdería precisión en los resultados, pero se mejoraría el tiempo de respuesta mostrando los resultados en el mapa de forma casi inmediata. Este modo ofrece otras desventajas al usuario, como son el consumo de batería y de datos de conexión al necesitar un funcionamiento constante. La elección de un período de actualización adecuado llevaría a una situación de compromiso entre la precisión de los resultados y la rapidez en obtenerlos y el gasto de batería y datos.

4.2.3 Posicionamiento en interiores

La última característica que diferencia la aplicación que se va a desarrollar con el resto de aplicaciones del mercado, es la capacidad de funcionar en recintos interiores o lugares donde no se dispone de cobertura GPS.

Para estos entornos se diseñará un sistema de posicionamiento basado en puntos de acceso inalámbricos WiFi. Este sistema de localización permitirá obtener la posición en función de las señales recibidas de los diferentes puntos de acceso inalámbricos.

En estos entornos con limitaciones de cobertura GPS es posible que también existan problemas de conectividad de red. Con el uso de APs WiFi se consigue a la vez dotar de conexión de red a los dispositivos móviles, la cual es necesaria también para el funcionamiento de la aplicación.

Durante el transcurso de este proyecto, a finales de 2011, Google Maps comenzó a dar servicio de posicionamiento en algunos recintos interiores como aeropuertos o centros comerciales de Estados Unidos. Esta característica no supone una barrera al proyecto, que va más allá, permitiendo la localización en cualquier recinto ad hoc sin cobertura GPS.

El sistema de posicionamiento para interiores, su diseño y demás características, se explican detalladamente en el siguiente capítulo.

Capítulo 5: Sistema de Posicionamiento

En este capítulo se estudiará en profundidad el sistema de posicionamiento para interiores, se verán algunas soluciones comerciales y se analizarán las diferentes técnicas, eligiendo y adaptando la que más se ajuste a la aplicación.

5.1 Introducción

El sistema de posicionamiento GPS está ampliamente extendido y proporciona un servicio preciso cuando se trata de espacios abiertos, pero en interiores, donde la recepción de la señal de los satélites no llega, hay que que recurrir a otras soluciones.

La opción planteada es disponer de una red de sensores inalámbricos y determinar la posición en función de la señal recibida de estos sensores. Hay distintos sistemas inalámbricos: bluetooth, Zigbee, etc, pero una solución práctica es utilizar puntos de acceso WIFI (802.11), ya que además permitirán dotar de conexión a Internet al espacio que se quiere cubrir.

5.2 Soluciones comerciales

Existen aplicaciones comerciales ya desarrolladas que dan un servicio de localización en interiores y que podrían servir para el fin que se persigue. Se verán las principales características de las aplicaciones y servicios más extendidas y su posible aplicación a este proyecto.

5.2.1 Ekahau

Ekahau [16] es una compañía que ofrece servicios de localización y seguimiento en tiempo real basados en redes Wi-Fi. Algunos de sus clientes son hospitales, empresas de logística y empresas con grandes flotas de vehículos, que utilizan este sistema para monitorizar las posiciones actuales de enfermos, mercancías y vehículos respectivamente.

El sistema de Ekahau opera con cualquier red Wi-Fi sin necesidad de infraestructura adicional, pudiendo monitorizar cualquier dispositivo con una interfaz 802.11. Para aquellos objetos que no cuentan con esta interfaz, dispone de etiquetas Wi-Fi que se añaden a los activos a seguir.

El sistema procesa la información recibida y a través de reglas de comportamiento permite lanzar alertas, generar informes o mostrar resultados en mapas. Se accede a los resultados a través de una aplicación web.

Ekahau cuenta con un cliente para *Android*, pero se trata de una aplicación propietaria, no un API que puede integrarse con otras aplicaciones. El precio del cliente es de 399 $.

El hecho de no poder integrar el cliente con nuestra propia aplicación hace que se descarte esta solución para este proyecto, además del coste elevado y de la necesidad de acceder a una web para obtener los resultados.

5.2.2 Skyhook

Skyhook [17] ofrece un sistema de posicionamiento para zonas urbanas y recintos interiores. Se basa en redes Wi-Fi y se destaca en sus características por la velocidad, fiabilidad y precisión. El sistema se basa en la información públicamente disponible, escaneando las redes Wi-Fi y asociándose con la posición en la que se detectan.

La gran ventaja que ofrece Skyhook es un SDK para *Android*, que permite acceder a su sistema desde aplicaciones de terceros.

Sin embargo, el gran inconveniente es la falta de información que presenta. Esta empresa nace en Estados Unidos, donde cuenta con abundante información de redes Wi-Fi, pero en España los datos no son tantos, sólo cuenta con cobertura en las principales zonas urbanas y no se referencian datos de recintos interiores como aeropuertos o centros comerciales. Además el hecho de no poder hacer uso del servicio en caso de un nuevo recinto o de disponer de una red ad hoc propia, hace que se descarte el uso de este sistema.

Figura 17: Cobertura red Skyhook en España[10]

[10] http://www.skyhookwireless.com/location-technology/coverage.php (Junio 2012)

5.2.3 Rosum

La empresa Rosum Corporation [18] desarrolló una serie de técnicas que aprovechan las señales de televisión para posicionar a los usuarios y dispositivos móviles, incluyendo las áreas interiores y urbanas, donde las tecnologías tradicionales de posicionamiento son menos eficaces.

Los dispositivos equipados con esta tecnología de posicionamiento son capaces de acceder a una serie de aplicaciones basadas en localización, que en lugar de utilizar los satélites GPS como plataformas de referencia, usa como balizas de referencia las antenas de televisión terrestres.

La tecnología de posicionamiento Rosum es un complemento para GPS ya que funciona en los entornos más urbanos donde el GPS tiende a fallar. Pero el problema que presenta este sistema, además de la necesidad de usar sus propias aplicaciones, es la imposibilidad de ser usado en lugares donde no hay cobertura. En ausencia de señales de televisión, como en un valle rodeado por montañas, el sistema de posicionamiento no funciona.

Por este motivo, este sistema no se ajusta a las necesidades de la aplicación que se pretende desarrollar.

5.2.4 Sonitor

El sistema de localización en tiempo real de Sonitor [19] utiliza etiquetas y receptores para localizar objetos y hacer un seguimiento en el interior de edificios. Las etiquetas transmiten una señal de identificación único por ondas de ultrasonido. Los receptores recogen la señal y la transmiten a través la red ya a un ordenador que la procesa.

Este sistema se diferencia de los anteriores en el uso de ultrasonidos, señales que no producen interferencias electromagnéticas con otros equipos, pero el uso de etiquetas y ultrasonidos no se adapta a las necesidades del sistema a diseñar.

5.2.5 WifiSlam

Wi-Fi Slam [20] es una aplicación para dispositivos móviles que permite obtener la ubicación es basándose en el conjunto de redes WiFi que le rodean. La herramienta obtiene la intensidad de las señales y los identificadores de las redes inalámbricas y los coteja con unos valores de referencia que obtiene de las bases de datos de WiFis de otras empresas como Google o Skyhook.

WifiSlam proporciona un SDK para integrar el sistema en otras aplicaciones, pero la gran desventaja de este sistema es que los datos de referencia tienen que ser recogidos de antemano. Este requisito hace que no sea un sistema adecuado y se descarte su uso.

5.2.6 Análisis

En los puntos anteriores se han analizado las características de algunos servicios comerciales de posicionamiento basado en redes Wi-Fi. El resultado que arroja este análisis es la

necesidad de implementar una solución propia, que se ajuste a las necesidades de la aplicación que se va a desarrollar, que resulte efectivo en entornos ad hoc y que no implique los elevados costes de las aplicaciones comerciales.

5.3 Técnicas de localización

El objetivo del sistema es obtener la posición a partir de un conjunto de puntos de acceso WiFi. Existen diferentes métodos que permiten calcular la posición a partir de las propiedades de las señales electromagnéticas que se reciben. A continuación se exponen algunas de las características de las señales y las posibles técnicas a utilizar.

5.3.1 RSSI

El parámetro RSSI (Received Signal Strength Indication) de una señal inalámbrica es el indicador de fuerza de señal recibida. Su uso para calcular distancias y posiciones se basa en la atenuación que sufre la señal a medida que se aleja de la fuente emisora. Calculando la atenuación sufrida por la señal en un punto, se puede calcular la distancia al punto de acceso.

El principal problema que presenta es que la atenuación es muy variable ya que depende de factores externos como reflexiones, interferencias, agentes atmosféricos,…

5.3.2 AOA

Son las siglas de Angle Of Arrival [21], esta técnica se basa en calcular la posición en función del ángulo de llegada de la señal.

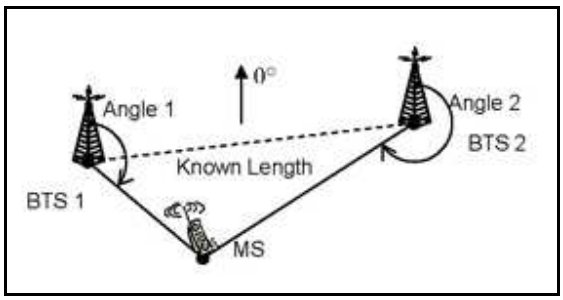

Figura 18: AOA[11]

El principal inconveniente que presenta esta técnica es que requiere un hardware especializado y visión directa entre emisor y receptor, situación que no siempre se podrá conseguir, por ejemplo en el caso de un recinto cerrado en el que la señal Wi-Fi llega a través de paredes. Por tanto, esta técnica no es aplicable para el sistema que se pretende diseñar.

[11] http://www.zess.uni-siegen.de/cms/upload/navigroup/Fig_12.jpg (Junio 2012)

5.3.3 TOA

Es una técnica basada en el tiempo de llegada (Time Of Arrival [22]) de una señal electromagnética. Existe una relación lineal entre el tiempo de propagación y la distancia entre un emisor y un receptor. Para calcular la distancia, será necesario que los relojes de los emisores y los receptores estén sincronizados y la distancia sea calculada a partir del tiempo de propagación desde que es emitida hasta que es recibida.

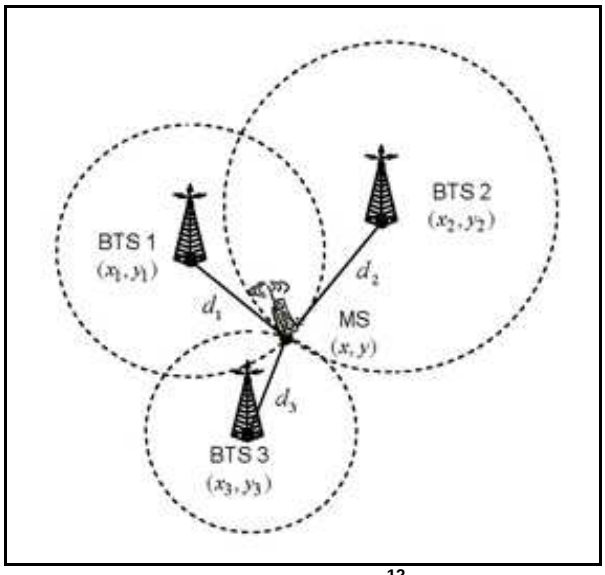

Figura 19: TOA[12]

Este sistema que requiere tener en cuenta los tiempos no se ajusta a nuestro modelo, ya que no será posible compartir dicha información.

5.3.4 TDOA

La técnica TDOA (Time Difference of Arrival o Diferencia de tiempo entre llegadas [21]) también conocida como posicionamiento hiperbólico, es un método en el que el receptor calcula la diferencia entre los tiempos de llegada de las señales de dos APs diferentes. Este método mejora al TOA en que no necesita los tiempos, solo sus diferencias. Pero tampoco se ajusta al modelo estudiado, en el que podemos tener solo uno o muchos emisores.

[12] http://www.zess.uni-siegen.de/cms/upload/navigroup/Fig_10.jpg (Junio 2012)

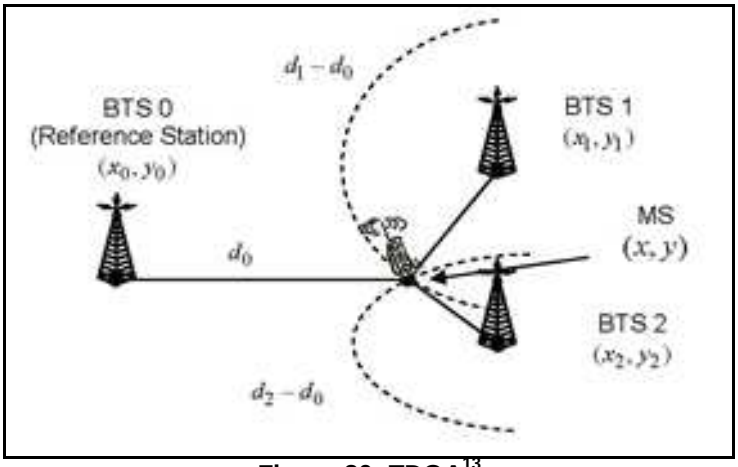

Figura 20: TDOA[13]

5.3.4 Modelos híbridos

Existen modelos híbridos que mejoran las técnicas TOA/AOA y que proporcionan mejores resultados en cuanto a precisión, pero estas técnicas presentan los mismos problemas que las técnicas originales, por lo que también se desechan.

5.3.5 Técnica elegida

De cara a diseñar el sistema de posicionamiento más adecuado para la aplicación, interesa el primer método, RSSI. Aunque presenta varios inconvenientes por los factores externos antes mencionados, que pueden hacer que esta técnica no parezca fiable a la hora de determinar una posición, es la opción elegida ya que es la más sencilla de implementar, no requiere hardware adicional y se adapta perfectamente a escenarios en los que no disponemos de información *a priori*. Asimismo no requiere cálculos previos, ya que es el propio dispositivo móvil el que calcula la potencia de la señal recibida y esta información está disponible a través del API de *Android*.

5.4 Localización mediante RSSI

El parámetro RSSI mide la intensidad de la señal recibida, la intensidad disminuye proporcionalmente a la distancia que recorre la señal electromagnética. Midiendo este parámetro podemos calcular la distancia recorrida por la señal.

Estas son algunas ventajas, inconvenientes y problemas que presenta este método:

Ventajas:

[13] http://www.zess.uni-siegen.de/cms/upload/navigroup/Fig_11.jpg (Junio 2012)

- Bajo coste, no requiere hardware adicional
- Bajo consumo de los dispositivos
- Se puede obtener esta información directamente en los dispositivos *Android*

Inconvenientes:
- Requiere una colocación adecuada de los puntos de acceso
- Si se requiere un despliegue denso de APs aumenta el coste
- Resultado muy dependiente del algoritmo utilizado.

Problemas en la medida del RSSI
- Reflexión
- Difracción
- Dispersión
- Interferencias

Los métodos que utilizan este parámetro se dividen en dos grupos: fingerprinting y trilateración.

5.4.1 Fingerprinting

La huella digital o *fingerprint []*, es un mapa de intensidades RSSI del área de interés. Una vez obtenido el mapa en la fase de entrenamiento, se puede proceder a la localización, para ello se lee el RSSI de todos los puntos de acceso visibles y se encuentra el punto que mejor coincide con el mapa trazado en la fase de entrenamiento. Los cambios en el ambiente pueden causar variaciones en la señal, de modo que las medidas tomadas pueden quedar obsoletas con el paso del tiempo.

El problema del método *fingerprint*, es que requiere una fase de entrenamiento, midiendo todas los valores RSSI de un determinado área. Se pretende realizar una aplicación que pueda ser usada en cualquier lugar, por tanto obtener un mapa de intensidades de todos los posibles lugares en que se use la aplicación, resulta inviable. Se descarta este método.

5.4.2 Trilateración

El método basado en trilateration o triangulación de potencia requiere conocer las coordenadas de los APs. A partir de las distancias calculadas a varios emisores, se pueden dibujar varios círculos, la intersección de estos círculos resulta en un punto el cual es la posición del objeto.

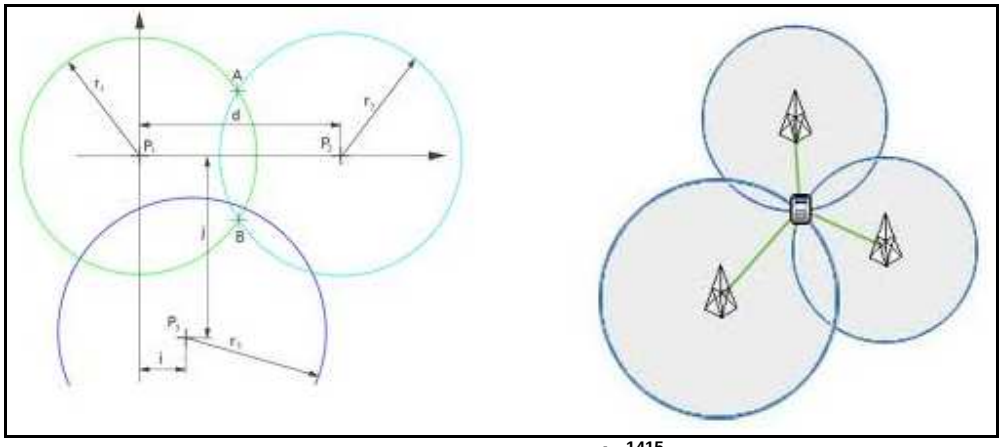

Figura 21: Trilateración[14][15]

La triangulación se basa en las siguientes fases:

- Con las potencias recibidas desde tres emisores se crea un sistema de ecuaciones, que representa tres círculos
- Se resuelve el sistema de ecuaciones para obtener un conjunto de puntos, llamados puntos de triangulación
- Cada punto de triangulación se considera el vértice de un triángulo
- Se forman todos los triángulos posibles y se calculan sus áreas para compararlas
- El centro del triángulo con el área más pequeña se toma como estimación de la ubicación del cliente

$$\rho_1 = \sqrt{(x_{e1} - x)^2 + (y_{e1} - y)^2}$$
$$\rho_2 = \sqrt{(x_{e2} - x)^2 + (y_{e2} - y)^2}$$
$$\rho_3 = \sqrt{(x_{e3} - x)^2 + (y_{e3} - y)^2}$$

Figura 22: Sistema de ecuaciones

Se descarta el método de triangulación de potencia porque para obtener la posición requiere comprobar los puntos en que se cortan los círculos de tres señales y a partir de ahí resolver un sistema de ecuaciones [23]. Esto supone un alto coste computacional tanto para un dispositivo móvil, con el consiguiente gasto energético, como para un servidor, en caso de tener gran número de ejecuciones.

5.4.3 Variación del método de trilateración

[14] http://es.wikipedia.org/wiki/Archivo:Trilateraci%C3%B3n.svg (Junio 2012)
[15] http://www.securityfocus.com/foundations/images/bt2-5.jpg (Junio 2012)

La aplicación de los métodos anteriores no resulta adecuada para el sistema de posicionamiento que se desea diseñar, pero sí es posible introducir variaciones para adaptarlo a los intereses de este proyecto.

Se opta por un sistema basado en trilateración, que a partir de la información de RSSI obtenida de varios APs y conocidas las posiciones de dichos APs, obtenga la posición del usuario. Los APs se identificarán por su dirección MAC, un identificador único para cada dispositivo de red.

El punto crítico, será obtener un algoritmo que sea capaz de calcular esta posición a partir de los datos anteriores con una precisión aceptable. Se probarán diferentes algoritmos, hasta encontrar un compromiso entre el cálculo que requieren y la precisión de los resultados obtenidos. La forma de medir la precisión será respecto a la posición obtenida por el GPS.

Para determinar el que mejor se adapta, se definirán varios posibles algoritmos, introduciendo variaciones sobre ellos. A través de unas aplicaciones de prueba, se estudiarán experimentalmente los diferentes algoritmos en un entorno controlado, analizando los resultados que producen. Se elegirá el que mejores resultados arroje para implementarlo en el sistema de posicionamiento.

5.5 Algoritmos

A partir del estudio de las pérdidas que sufre una señal electromagnética al propagarse, bajo una serie de suposiciones, definiremos un algoritmo para obtener la posición en función de las potencias (RSSI) de las señales y de las posiciones de los APS. La idea es obtener la posición del usuario como una combinación lineal de las posiciones de los APs, con unos pesos de cada factor proporcionales a la potencia recibida de cada AP. La función del algoritmo será convertir los valores RSSI de cada AP en los pesos de la ecuación.

5.5.1 Algoritmo base

El algoritmo en que basaremos la obtención de la posición, y sobre el que posteriormente se introducirán variaciones, se basa en la atenuación que sufre una señal propagándose. Las pérdidas que sufre una onda electromagnética en espacio libre vienen determinadas por la ecuación:

$$l_{bf} = \left(\frac{4\pi d}{\lambda}\right)^2$$

Dónde d = distancia al emisor y λ = longitud de onda

Considerando estas pérdidas, la potencia de la señal disminuye con el cuadrado de la distancia que recorre:

$$P_{rx} = \frac{P_{tx}}{L_{bf}} = \frac{P_{tx}\cdot\lambda^2}{(4\pi d)^2} = \kappa\cdot\frac{1}{d^2} \Rightarrow \sqrt{P_{rx}}\ \alpha\ \frac{1}{d}$$

Dónde K = constante

Si se pretende conocer la distancia recorrida por la señal en un punto, se deduce que es inversamente proporcional a la raíz cuadrada de la potencia recibida en ese punto.

A continuación se normalizan los valores de las potencias recibidas de cada AP, para hacer comparables todos los valores. La normalización se produce dividiendo cada potencia recibida entre el valor mayor de todas las potencias recibidas.

$$P_{i_{norm}} = \frac{P_i}{P_{max}}$$

Y son las raíces de estas potencias normalizadas las que se utilizan como pesos en la ecuación para determinar la posición.

$$W_i = \sqrt{P_{i_{norm}}}$$

La posición se obtiene finalmente como una combinación lineal de las posiciones de los APs, en la que los pesos o coeficientes de cada término son los pesos anteriormente calculados.

$$X = \frac{1}{N} \cdot \sum_{i=1}^{N} W_i \cdot X_i$$

Dónde Xi = posición de cada AP, N = número de APs y X = posición del usuario

Si conocemos de cada AP sus coordenadas de longitud y latitud, para obtener las coordenadas del usuario, la fórmula aplicada sería la misma:

$$Long = \frac{1}{N} \cdot \sum_{i=1}^{N} W_i \cdot Long_i$$

$$Lat = \frac{1}{N} \cdot \sum_{i=1}^{N} W_i \cdot Lat_i$$

Dónde Longi = coordenada de longitud de cada AP, Lati = coordenada de latitud de cada AP, N = número de APs y X = posición del usuario

Para los cálculos anteriores, se consideran una serie de suposiciones:

- Antenas omnidireccionales
- Misma potencia emitida por cada antena (AP)
- Misma frecuencia
- Solo consideramos pérdidas en espacio libre

5.5.2 Variaciones

Al tratarse de un algoritmo del que habrá que probar su eficiencia o precisión, se probará a definir nuevos algoritmos, introduciendo una o más pequeñas variaciones sobre el algoritmo base anterior, con el objetivo de comprobar si ofrecen mejores resultados.

5.5.2.1 Normalización

Una primera variación consistirá en no normalizar los datos de las potencias recibidas. La normalización se realiza para hacer que los datos sean comparables, pero al tratarse de potencias que provienen de APs de similares características, los datos pueden ser ya comparables.

5.5.2.2 Datos atípicos

Otra variación será, en caso de tener varios APs, eliminar la información de aquellos que se alejen mucho de los valores del resto. El objetivo de eliminar datos aislados o infrecuentes, que pueden ser erróneos por alguno de los factores externos que interfieren, es evitar que alteren la precisión del resto de los datos.

5.5.2.3 Raíces

Se relaciona la potencia con el cuadrado de la distancia, por ello al operar, se toman las raíces cuadradas de los pesos. Pero experimentalmente se puede probar a no tomar raíces para comprobar los resultados teóricos.

5.5.2.4 Unidades

Si los datos de potencias se obtienen en unidades logarítmicas (dBm) se puede operar en unidades logarítmicas o convertir los datos a unidades naturales, al haber operaciones no lineales en el proceso, puede haber cierta variación operando en unas unidades u otras. Otra posible variación en los algoritmos es operar en diferentes unidades.

Capítulo 6: Aplicaciones de experimentación

Se analizan y diseñan en este apartado un conjunto de aplicaciones para dispositivos *Android* cuyo objetivo es obtener la información necesaria que permita decidir qué algoritmo usar para calcular la posición en la que se encuentra un usuario, conocidas las posiciones de varios puntos de acceso WIFI y las potencias con que se recibe la señal de cada punto de acceso.

6.1 Aplicación PosiciónGPS

El objeto de esta aplicación es determinar la posición GPS actual. El propósito es obtener las posiciones GPS de los distintos puntos de acceso WIFI que servirán de referencia para determinar posteriormente la posición GPS del usuario.

La aplicación es sencilla y los requisitos son muy básicos, su funcionalidad básica debe ser mostrar por pantalla la posición GPS actual, es decir, las coordenadas de longitud y latitud del punto en el que se encuentra el dispositivo móvil.

6.1.1 Casos de uso

Como la aplicación tiene un único propósito, solo ejecuta una acción, contemplamos un único caso de uso:

Identificación CU	*CU-1*
Nombre CU	Obtener la posición GPS actual
Propósito CU	Ejecutamos la aplicación y pulsamos el botón de buscar, la aplicación obtendrá la posición GPS y mostrará la información obtenida por pantalla.
Actores involucrados	Usuario
Secuencia operativa	1. Ejecutar la aplicación. 2. Pulsar botón “buscar”. 3. Mostrar mensaje mientras se realiza la búsqueda. 4. Mostrar información por pantalla.

	Cursos Alternativos: 5. Línea 3: En caso de no estar activada la conexión GPS se mostrará un mensaje informando al usuario. 6. Línea 4: Si el usuario pulsa el botón de volver/cancelar se cancela la búsqueda y se vuelve a la pantalla inicial.

Proceso 7: PosicionGPS Caso de uso CU-1

6.1.2 Requisitos funcionales

Los requisitos funcionales son una descripción de alto nivel del servicio, muestran las características requerida del sistema y expresan las acción que el sistema debe realizar.

Código Requisito	CR-1
Prioridad	Alta
Nombre de requisito	Interfaz de entrada
Descripción de requisito	Se muestra al arrancar, información sobre el funcionamiento de la propia aplicación.

Requisito 8: PosicionGPS CR-1

Código Requisito	CR-2
Prioridad	Alta
Nombre de requisito	Conexión GPS
Descripción de requisito	La aplicación deberá avisar al usuario si la conexión GPS está activa en el momento de realizar la búsqueda de la posición.

Requisito 9: PosicionGPS CR-2

Código Requisito	CR-3
Prioridad	Alta
Nombre de requisito	Búsqueda
Descripción de requisito	La aplicación deberá mostrar un mensaje al usuario mientras se realiza la búsqueda.

Requisito 10: PosicionGPS CR-3

Código Requisito	CR-4
Prioridad	Alta
Nombre de requisito	Resultado
Descripción de requisito	El resultado de la búsqueda (las coordenadas de longitud y latitud de la posición GPS actual) se muestra en la pantalla.

Requisito 11: PosicionGPS CR-4

Código Requisito	CR-5
Prioridad	Alta
Nombre de requisito	Cancelar búsquedas
Descripción de requisito	En cualquier momento el usuario puede cancelar la búsqueda volviendo a la pantalla anterior.

Requisito 12: PosicionGPS CR-5

6.1.3 Requisitos no funcionales

Los requisitos no funcionales son característica requerida del sistema o del servicio que detallan restricciones de las funciones del sistema, tales como restricciones de tiempo, estándares, etc.

Código del requisito	Prioridad	Descripción del requisito
CR-6	**Alta**	La aplicación funcionará sobre un sistema operativo Android 2.2 o superior.
CR-7	**Alta**	Es necesario para el funcionamiento de la aplicación que el dispositivo sobre el que se ejecute disponga de chip GPS.

Requisito 13: PosicionGPS Requisitos No Funcionales

6.1.4 Pruebas

Se diseña un conjunto de pruebas para la aplicación que permitan verificar los requisitos funcionales. Las pruebas son las siguientes:

Código Prueba	P-1
Requisito asociado	CR-1, CR-2
Descripción	Se ejecuta la aplicación sin activar la conexión GPS. Se pulsa el botón para buscar.
Resultado	Se muestra un mensaje informando que la conexión está desactivada.

Prueba 1: PosicionGPS P-1

Código Prueba	P-2
Requisito asociado	CR-1, CR-3
Descripción	Se ejecuta la aplicación con la conexión GPS activada. Se presiona el botón de buscar.
Resultado	Se muestra un mensaje informando que se esta realizando la búsqueda.

Prueba 2: PosicionGPS P-2

Código Prueba	P-3
Requisito asociado	CR-1, CR-3, CR-5
Descripción	Se ejecuta la aplicación con la conexión GPS activada. Se presiona el botón de buscar. Mientras se realiza la búsqueda se presiona el botón de cancelar.
Resultado	Al presionar el botón de cancelar, se vuelve a la pantalla anterior.

Prueba 3: PosicionGPS P-3

Código Prueba	P-4
Requisito asociado	CR-1, CR-3, CR-4
Descripción	Se ejecutamos la aplicación con la conexión GPS activada. Se

	presiona el botón de buscar y se espera a que la búsqueda se complete.
Resultado	Al finalizar se muestran en al pantalla las coordenadas de la posición actual.

Prueba 4: PosicionGPS P-4

6.1.4.1 Matriz de trazabilidad

Matriz que relaciona los requisitos con las pruebas que los verifican.

Requisito\Prueba	P-1	P-2	P-3	P-4
CR-1	X	X	X	X
CR-2	X			
CR-3		X	X	X
CR-4				X
CR-5			X	

Tabla 3: PosicionGPS Matriz de trazabilidad Requisitos - Pruebas

6.1.5 Diseño de interfaces

El diseño de las interfaces de la aplicación es sencillo, dado que las funcionalidades requeridas son también sencillas. La interfaz de usuario únicamente necesita una pantalla con la información de la aplicación, con un botón para que lance las búsquedas y con los campos de longitud y latitud en los que se actualizan los resultados de la nueva búsqueda.

Como se trata de una aplicación que nos servirá para toma de datos y realizar pruebas, no es una aplicación que utilizarán los usuarios finales, no se ha puesto especial empeño en el diseño gráfico.

A continuación se muestran las pantallas de la aplicación:

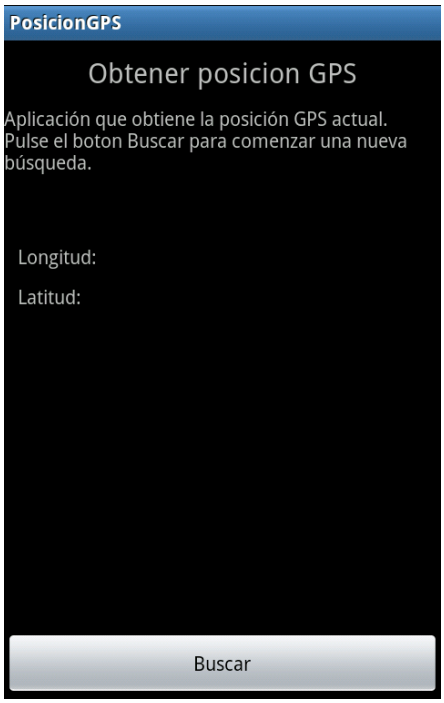

Figura 23: Pantalla inicial.

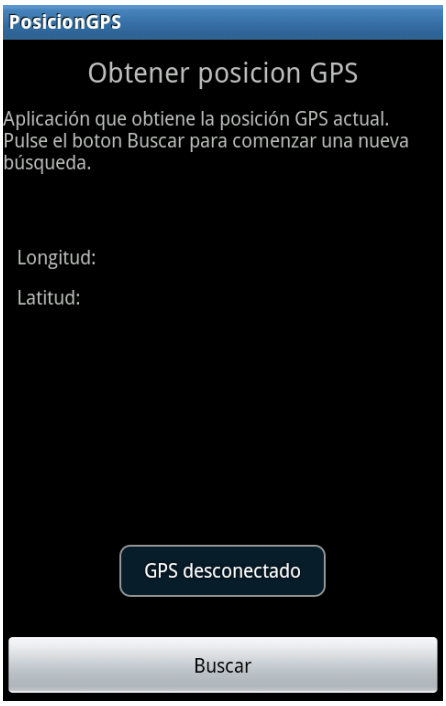

Figura 24: Pantalla GPS OFF

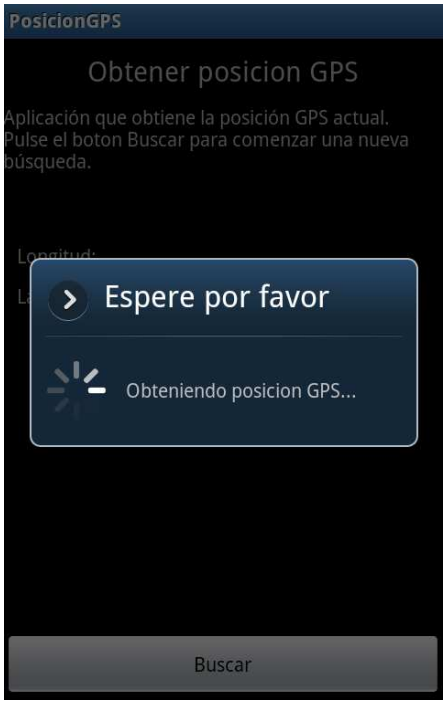

Figura 25: Pantalla de búsqueda.

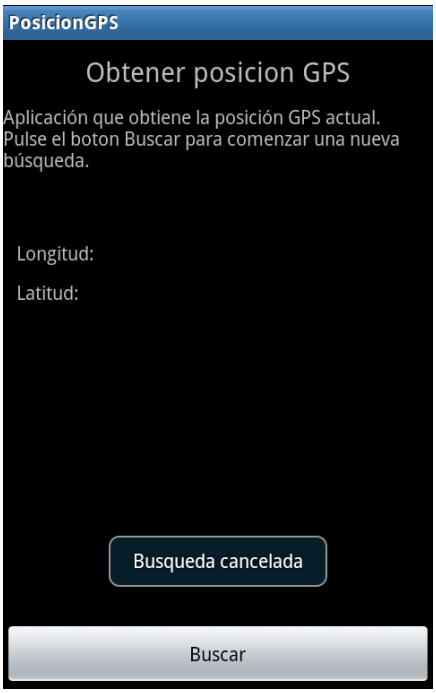

Figura 26: Pantalla de cancelación

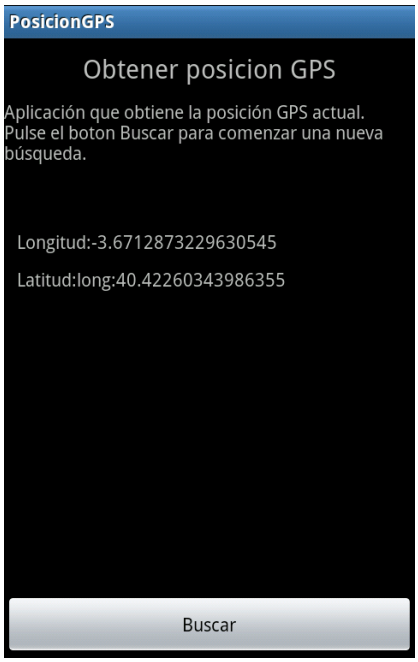

Figura 27: Pantalla de resultado.

6.1.6 Diagrama de clases

Debido a la simplicidad de la aplicación no es necesario un diagrama de componentes complejo, ya que toda la lógica se ejecuta en una única clase:

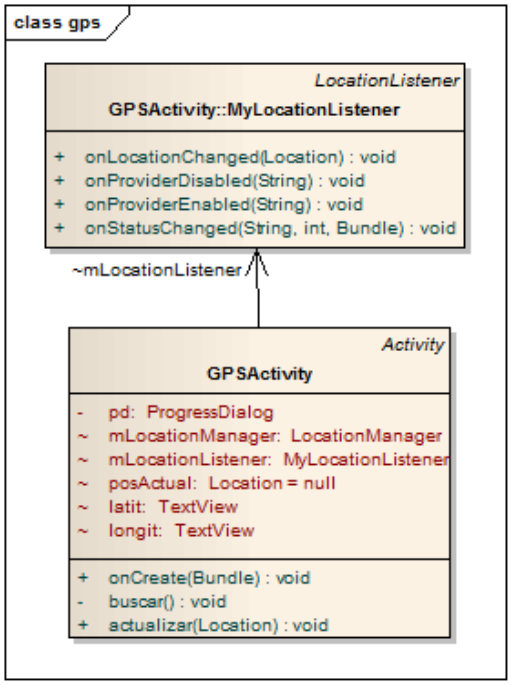

Figura 28: Diagrama de clases

6.1.7 Diagrama de estados

A través del diagrama de estados se describe gráficamente el comportamiento del sistema y se muestran todos los posibles estados por los que puede pasar.

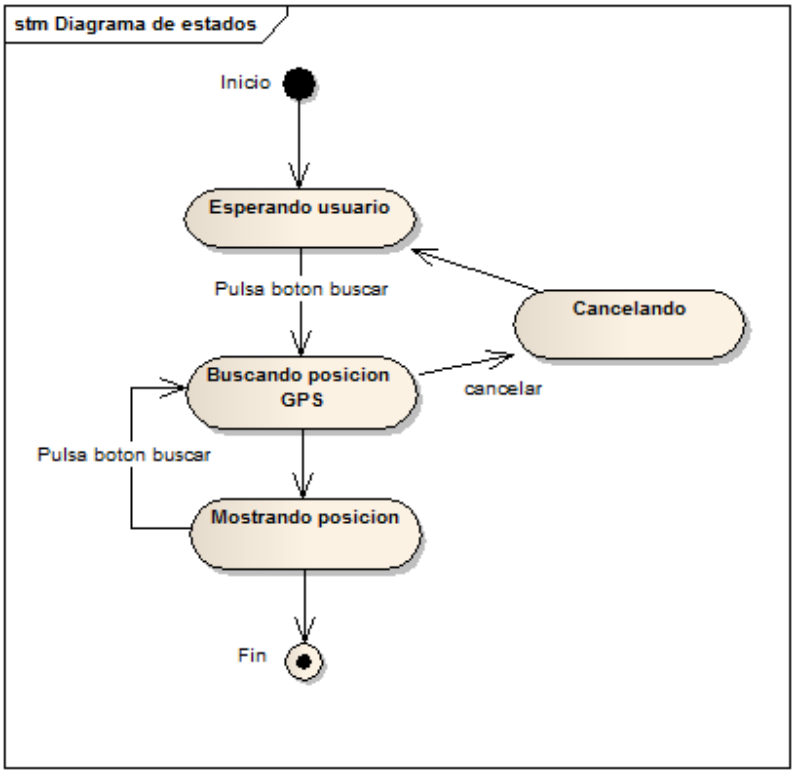

Figura 29: Diagrama de estados

6.1.8 Implementación

La implementación de esta aplicación es de complejidad baja. Además de implementarla para llevar a cabo las pruebas, permite adquirir los conocimientos necesarios sobre la implementación haciendo uso del GPS para aplicarlos tanto en la siguiente aplicación que se desarrollará, la aplicación PosiciónWifi, como en la aplicación final.

6.1.8.1 Pruebas verificadas

Tras la implementación se verifica el conjunto de pruebas, los resultados obtenidos son:

Prueba	Estado
P-1	Verificada
P-2	Verificada
P-3	Verificada
P-4	Verificada

Tabla 4: PosicionGPS Pruebas Verificadas

6.2 Aplicación PosiciónWiFi

La aplicación *PosicionWiFi* nos permite estimar la posición en que se encuentra el usuario a partir de las posiciones conocidas de los puntos de acceso WiFi cercanos. El objeto de la aplicación es calcular a partir de estas posiciones conocidas y las potencias recibidas de cada punto de acceso la posición del usuario.

Para determinar la posición con los datos obtenidos, se pueden seleccionar diferentes algoritmos para la estimación. La aplicación permite seleccionar varios algoritmos y calcular la posición con cada uno de ellos. Finalmente a través de la señal GPS del dispositivo se obtiene la posición real y se compara con cada una de las calculadas, mostrando el margen de error asociado a cada algoritmo.

6.2.1 Casos de uso

Como la aplicación tiene un único propósito, solo ejecuta una acción principal, contemplamos un único caso de uso:

Identificación CU	*CU-1*
Nombre CU	Obtener posiciones estimadas
Propósito CU	Ejecutamos la aplicación, desde la pantalla inicial podemos configurar los distintos algoritmos que vamos a comparar para obtener la posición actual. Presionando el botón buscar nos muestra una lista con los puntos de acceso Wifi geolocalizados. Desde esa pantalla presionamos el botón localizar y nos muestra una lista con los algoritmos seleccionados y sus márgenes de error respecto a la posición GPS real.
Actores involucrados	Usuario

Secuencia operativa	1. Ejecutar la aplicación. 2. Pulsar botón “buscar”. 3. Mostrar mensaje mientras se realiza la búsqueda. 4. Mostrar información de Wifis por pantalla. 5. Pulsar botón “localizar”. 6. Mostrar mensaje mientras se realiza la búsqueda. 7. Mostrar información de Algoritmos por pantalla. **Cursos Alternativos:** 8. Línea 3: En caso de no estar activada la conexión WIFI se mostrará un mensaje informando al usuario. 9. Línea 6: Si no hay ninguna red disponible se informa al usuario y no se permite continuar 10. Línea 6: En caso de no estar activada la conexión GPS se mostrará un mensaje informando al usuario.

Proceso 8: PosicionWiFi Caso de uso CU-1

6.2.2 Requisitos funcionales

Los requisitos funcionales son una descripción de alto nivel del servicio, muestran las características requerida del sistema y expresan las acción que el sistema debe realizar.

Código Requisito	CR-1
Prioridad	Alta
Nombre de requisito	Interfaz de entrada
Descripción de requisito	Se muestra al arrancar, información sobre el funcionamiento de la propia aplicación.

Requisito 14: PosicionWiFi CR-1

Código Requisito	CR-2
Prioridad	Alta
Nombre de requisito	Selección de algoritmos
Descripción de requisito	Se mostrará una lista de los algoritmos disponibles con su descripción, que seleccionarlos y deseleccionarlos.

Requisito 15: PosicionWiFi CR-2

Código Requisito	CR-3
Prioridad	Alta
Nombre de requisito	Conexión WIFI
Descripción de requisito	La aplicación deberá avisar al usuario si la conexión Wifi está activa en el momento de realizar la búsqueda de la posición.

Requisito 16: PosicionWiFi CR-3

Código Requisito	CR-4
Prioridad	Alta
Nombre de requisito	Búsqueda WIFI
Descripción de requisito	La aplicación deberá mostrar un mensaje al usuario mientras se realiza la búsqueda.

Requisito 17: PosicionWiFi CR-4

Código Requisito	CR-5
Prioridad	Alta
Nombre de requisito	Mostrar redes Wifi
Descripción de requisito	Antes de determinar la posición, se listarán las redes wifi disponibles, es decir, aquellas de las que se conoce la posición GPS. Si no se obtiene ninguna, no se podrá continuar con la ejecución.

Requisito 18: PosicionWiFi CR-5

Código Requisito	CR-6
Prioridad	Alta
Nombre de	Informacion red WiFi

requisito	
Descripción de requisito	Cuando el usuario pulsa sobre una de las redes WiFi de la lista, se muestra información detallada de la misma: SSID, MAC, potencia, posición, etc.

Requisito 19: PosicionWiFi CR-6

Código Requisito	CR-7
Prioridad	Alta
Nombre de requisito	Conexión GPS
Descripción de requisito	La aplicación deberá avisar al usuario si la conexión GPS está activa en el momento de realizar la búsqueda de la posición.

Requisito 20: PosicionWiFi CR-7

Código Requisito	CR-8
Prioridad	Alta
Nombre de requisito	Búsqueda GPS
Descripción de requisito	La aplicación deberá mostrar un mensaje al usuario mientras se realiza la búsqueda.

Requisito 21: PosicionWiFi CR-8

Código Requisito	CR-9
Prioridad	Alta
Nombre de requisito	Resultado
Descripción de requisito	Tras obtener la posición GPS, las posiciones calculadas con los algoritmos seleccionados y compararlos, se mostrará una lista con los resultados de las comparaciones.

Requisito 22: PosicionWiFi CR-9

Código Requisito	CR-10
Prioridad	Alta
Nombre de requisito	Cancelar búsquedas
Descripción de requisito	En cualquier momento el usuario puede cancelar la búsqueda volviendo a la pantalla anterior.

Requisito 23: PosicionWiFi CR-10

6.2.3 Requisitos no funcionales

Los requisitos no funcionales que se definen a continuación, son características requeridas del sistema o del servicio que detallan restricciones de las funciones del sistema, tales como restricciones de tiempo, estándares, etc.

Código requisito	***Prioridad***	***Descripción del requisito***
CR-11	**Alta**	La aplicación funcionará sobre un sistema operativo Android 2.2 o superior.
CR-12	**Alta**	El dispositivo sobre el que se ejecuta la aplicación debe disponer de conexión de red inalámbrica (WIFI).
CR-13	**Alta**	El dispositivo sobre el que se ejecuta la aplicación debe disponer de chip GPS.
CR -14	**Alta**	La información de la posición GPS de los puntos de acceso WIFI (longitud y latitud) se almacena en un fichero de configuración externo a la aplicación, en el sistema de ficheros del dispositivo
CR-15	**Alta**	No se permitirá que la pantalla al girar cambie de modo *portrait* a *landscape*, para evitar que se recarguen los datos de la aplicación.

Requisito 24: PosicionWiFi Requisitos no funcionales

6.2.4 Pruebas

Se diseña un conjunto de pruebas en la aplicación que permitan verificar los requisitos funcionales de la aplicación. Las pruebas son las siguientes:

Código Prueba	P-1
Requisito asociado	CR-1, CR-3
Descripción	Se accede a la aplicación con la conexión WIFI desactivada. Se pulsa el botón de buscar para comenzar la ejecución.
Resultado	Se muestra un mensaje informando que la conexión está desactivada.

Prueba 5: PosicionWiFi P-1

Código Prueba	P-2
Requisito asociado	CR-1, CR-3, CR-4, CR-5
Descripción	Se ejecuta la aplicación con la conexión WIFI activada. Se presiona el botón de buscar.
Resultado	Se muestra un mensaje informando que se esta realizando la búsqueda. A continuación una lista con las redes WIFI detectadas de las que conocemos su posición.

Prueba 6: PosicionWiFi P-2

Código Prueba	P-3
Requisito asociado	CR-1, CR-3, CR-4, CR-10
Descripción	Se ejecuta la aplicación con la conexión WIFI activada. Se presiona el botón de buscar. Mientras se realiza la búsqueda se presiona el botón de cancelar.
Resultado	Se muestra un mensaje informando que se esta realizando la búsqueda. Al presionar el botón de cancelar, se vuelve a la pantalla anterior.

Prueba 7: PosicionWiFi P-3

Código Prueba	P-4
Requisito asociado	CR-1, CR-3, CR-4, CR-5
Descripción	Se ejecuta la aplicación con la conexión WIFI activada. Se presiona el botón de buscar en un punto en el que no hay al alcance ninguna red geolocalizada.
Resultado	Se muestra un mensaje informando que se esta realizando la búsqueda. A continuación la lista con las redes WIFI detectadas vacía. Si se presiona el botón de localizar se muestra un mensaje de error.

Prueba 8: PosicionWiFi P-4

Código Prueba	P-5
Requisito asociado	CR-7
Descripción	Desde la pantalla de la lista de redes WIFI, con la conexión GPS desactivada, se presiona el botón de localizar.
Resultado	Se muestra un mensaje informando que la conexión está desactivada.

Prueba 9: PosicionWiFi P-5

Código Prueba	P-6
Requisito asociado	CR-6, CR-7, CR-8, CR-10
Descripción	Desde la pantalla de la lista de redes WIFI, con la conexión GPS activada, se presiona el botón de localizar. Mientras se realiza la búsqueda se presiona el botón de cancelar. Se pincha sobre una de las WiFis.
Resultado	Se muestra un mensaje informando que la búsqueda se está realizando. Al presionar el botón de cancelar, se vuelve a la pantalla anterior. Al pinchar sobre una WiFi muestra la información asociada.

Prueba 10: PosicionWiFi P-6

Código Prueba	P-7
Requisito asociado	CR-7, CR-8, CR-9
Descripción	Desde la pantalla de la lista de redes WIFI, con la conexión GPS activada, se presiona el botón de localizar.
Resultado	Se muestra un mensaje informando que la búsqueda se está realizando. Al finalizar, se muestra una lista con los resultados de margen de error en el cálculo de la posición de cada algoritmo.

Prueba 11: PosicionWiFi P-7

Código Prueba	P-8
Requisito asociado	CR-2, CR-7, CR-8, CR-9
Descripción	Desde la pantalla inicial, se presiona el botón configurar y se selecciona un subconjunto de algoritmos. Se vuelve a la pantalla inicial. Resto de ejecución normal.
Resultado	Se muestra a lista con los algoritmos y sus descripciones, al pinchar en algunos se van deseleccionando. Al finalizar la ejecución, se muestra una lista con los resultados de margen de error en el cálculo de la posición de cada algoritmo que estaba seleccionado.

Prueba 12: PosicionWiFi P-8

6.2.4.1 Matriz de trazabilidad

Matriz que relaciona los requisitos con las pruebas que los verifican.

Requisito\Prueba	P-1	P-2	P-3	P-4	P-5	P-6	P-7	P-8
CR-1	X	X	X	X				
CR-2								X
CR-3	X	X	X	X				
CR-4		X	X	X				
CR-5		X		X				

CR-6						X		
CR-7					X	X	X	X
CR-8						X	X	X
CR-9							X	X
CR-10			X			X		

Tabla 5: PosicionWiFi Matriz de trazabilidad Requisitos - Pruebas

6.2.5 Diseño de interfaces

El diseño de la aplicación se ha hecho buscando la sencillez, dado que las funcionalidades requeridas son también de complejidad no elevada.

A través de 4 pantallas diferentes se puede mostrar toda la información requerida por la aplicación y como se trata de una aplicación que servirá para toma de datos y pruebas (no es una aplicación que utilizarán otros usuarios) no se ha puesto especial empeño en el diseño gráfico. A continuación se muestran las pantallas de la aplicación:

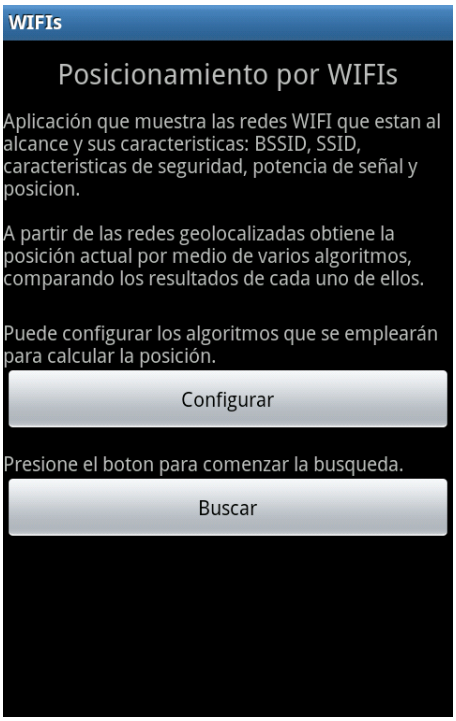

Figura 30: Pantalla inicial (P_ini)

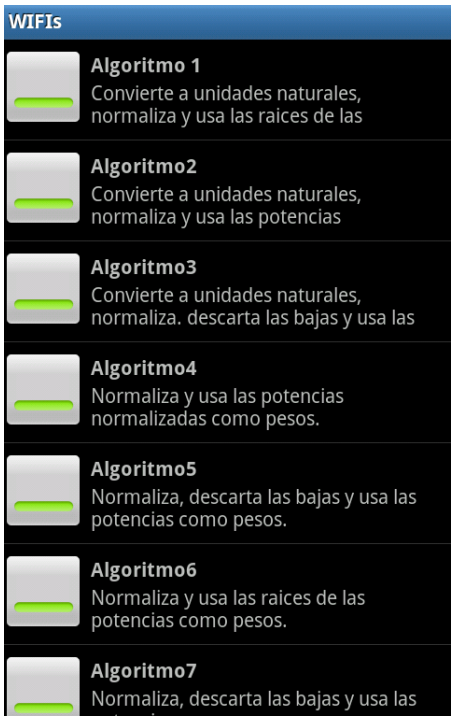

Figura 31: Pantalla selección de algoritmos (P_alg)

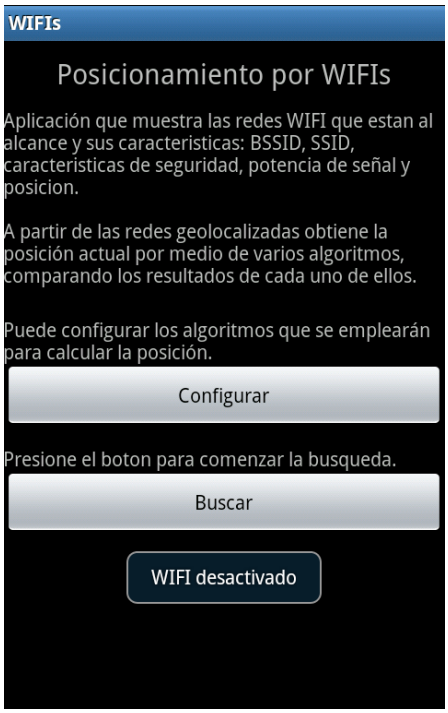

Figura 32: Pantalla WIFI OFF (P_Woff)

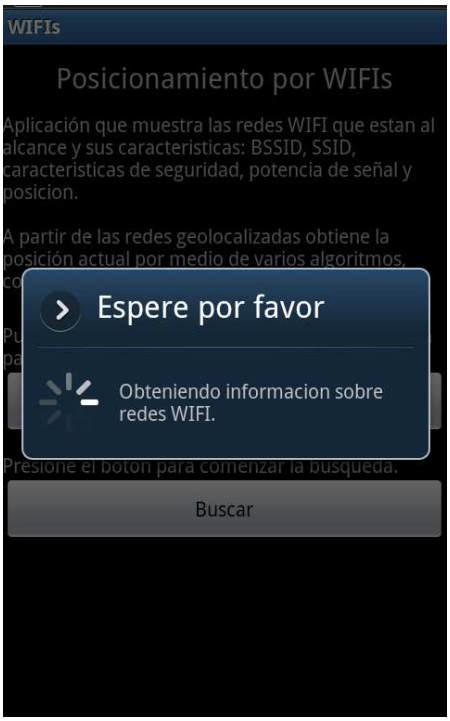

Figura 33: Pantalla búsqueda WiFis. (P_Wsearch)

Figura 34: Pantalla lista Wifis (P_Wifis)

Figura 35: Pantalla información AP (P_Winfo)

Figura 36: Pantalla GPS OFF. (P_Goff)

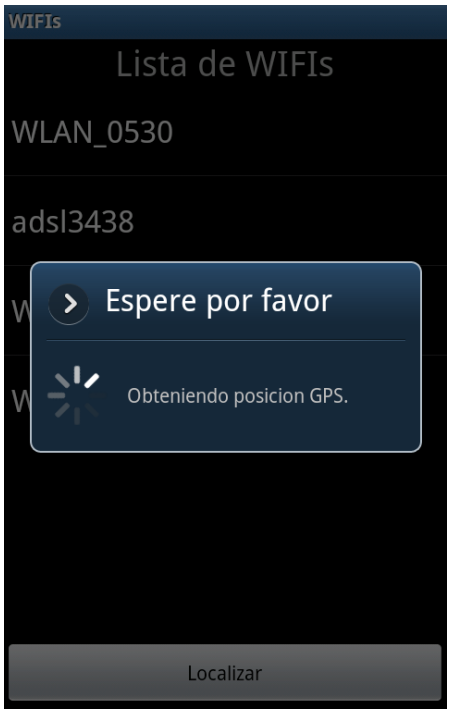

Figura 37: Pantalla búsqueda GPS. (P_Gsearch)

Figura 38: Pantalla con resultados en metros (P_res)

6.2.6 Diagrama de estados

A través del diagrama de estados se describe gráficamente el comportamiento del sistema y se muestran todos los posibles estados por los que puede pasar. Se parte de la acción de arrancar al aplicación y se muestran todas las transiciones y desviaciones hasta alcanzar el punto final en que se obtienen los resultados.

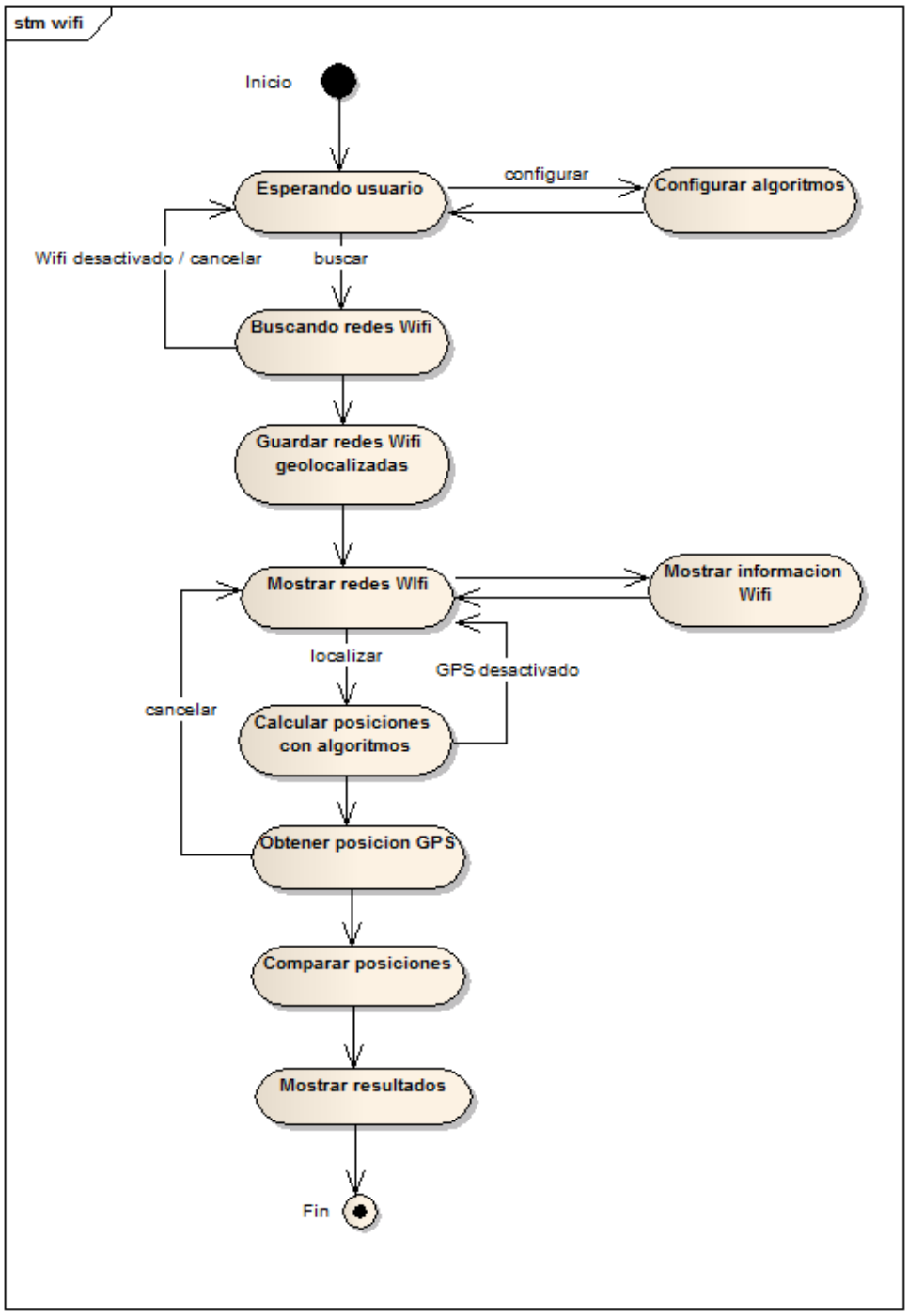

Figura 39: Diagrama de estados

6.2.7 Diagrama de navegación

En el diagrama de navegación de la Figura 40 se representan las distintas transiciones entre las pantallas de la aplicación.

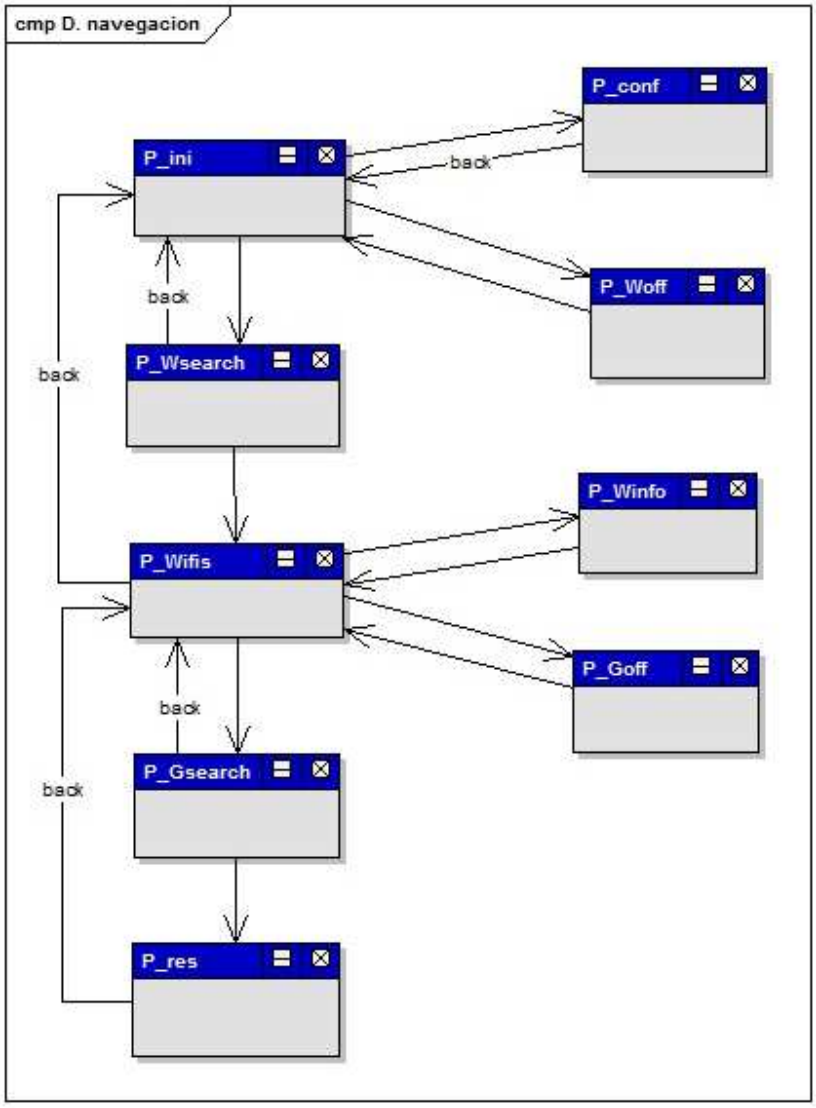

Figura 40: Diagrama de navegación

6.2.8 Diagrama de clases

Android utiliza el patrón de diseño MVC [24], cuya principal ventaja consiste en separar los datos de una aplicación, la interfaz de usuario y la lógica de negocios en tres componentes distintos

- Modelo. Representa el conocimiento, se refiere a las representaciones que se construyen basadas en la información. Es la representación de los datos que maneja la

aplicación, son clases Java. En las Figuras 41 y 42 se ven representan las clases que componen el modelo.

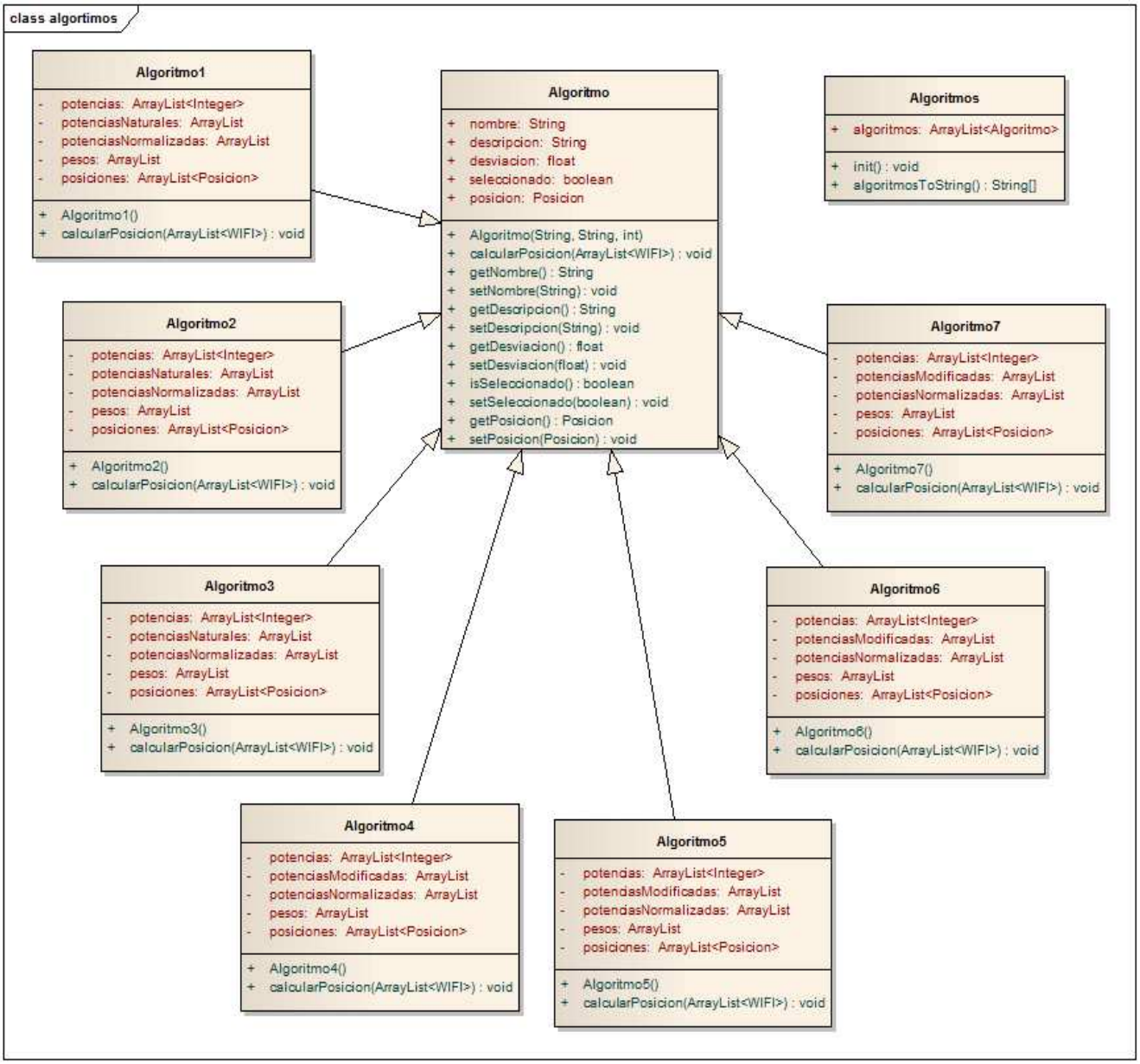

Figura 41: Modelo - Algoritmos

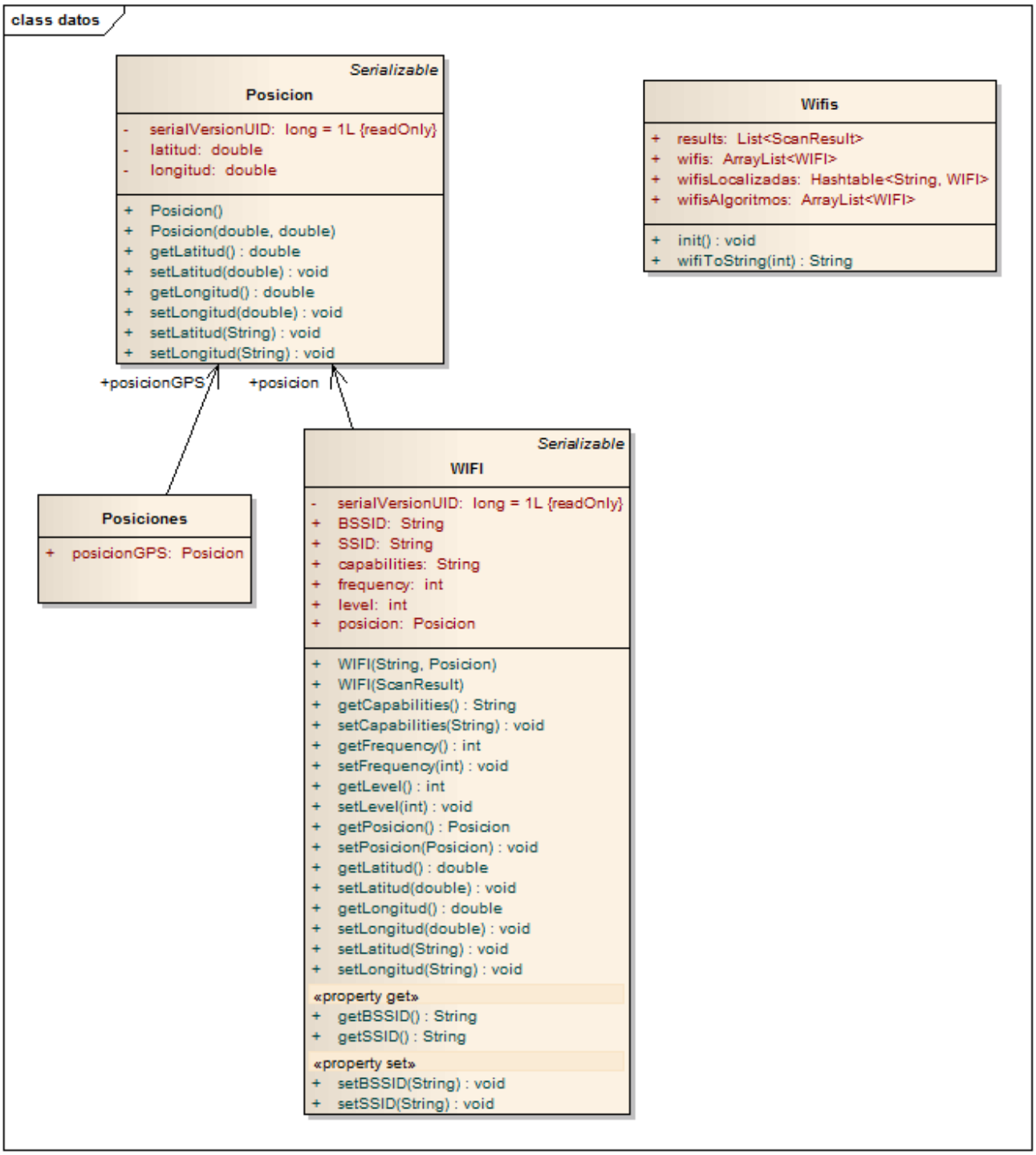

Figura 42: Modelo - Datos

- Vista. La vista es una representación visual del modelo, es la interfaz con la que va a interactuar el usuario. En Android, las interfaces se construyen con archivos XML.

- Controlador. El controlador relaciona al usuario y al sistema, son todas esas clases que dan vida a las interfaces y que permitirán desplegar y consumir información de/para el usuario. Estos controladores se programan en lenguaje Java y son el core de la aplicación, se denominan Actividades y se pueden ver en la siguiente Figura:

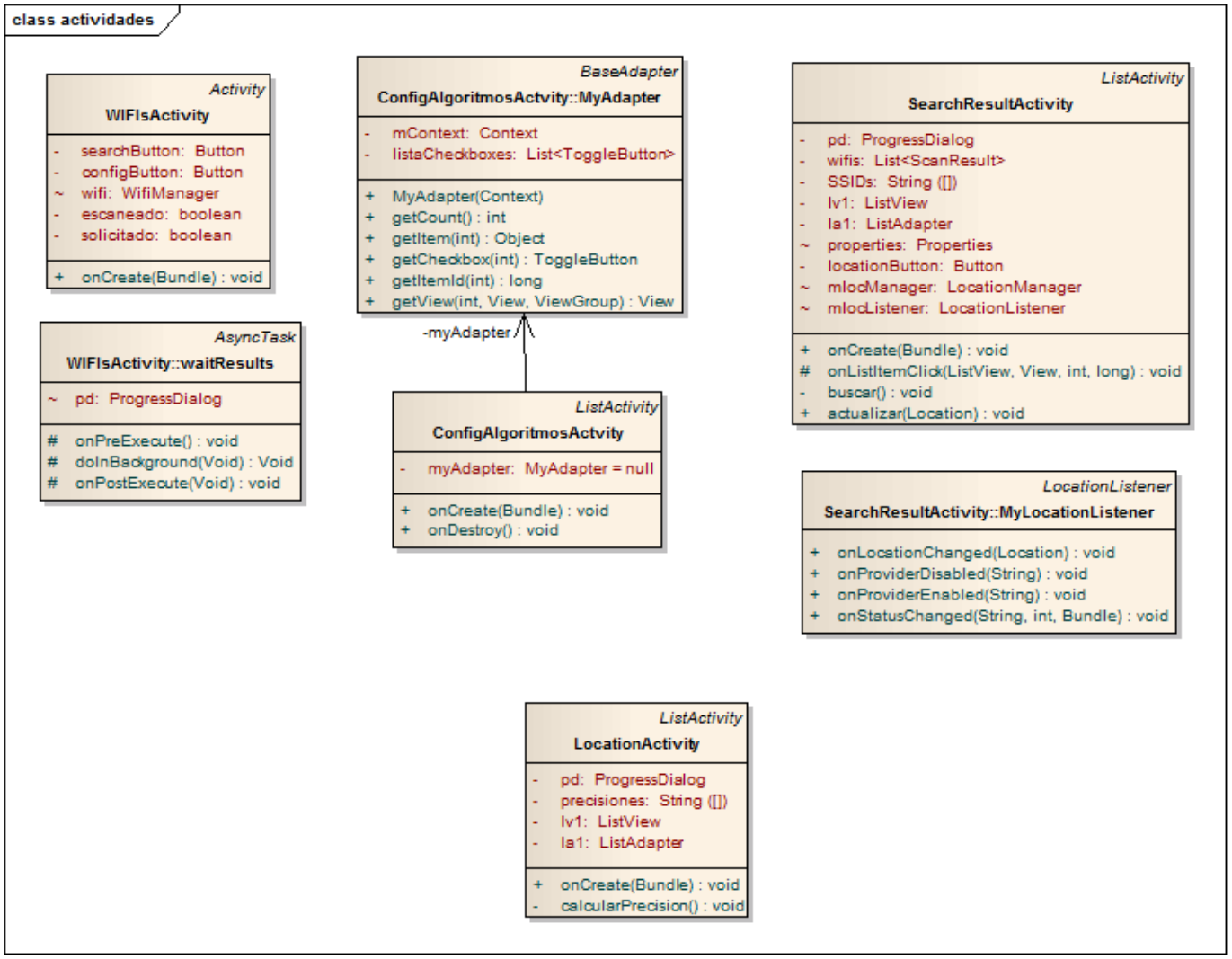

Figura 43: Controlador - Actividades

6.2.9 Diagrama de secuencia

Se muestran en las Figuras siguientes varios diagramas de secuencia que representan diferentes ejecuciones.

- Selección de algoritmos: Se representa la secuencia de llamadas para cambiar la configuración de los algoritmos seleccionados para la ejecución.

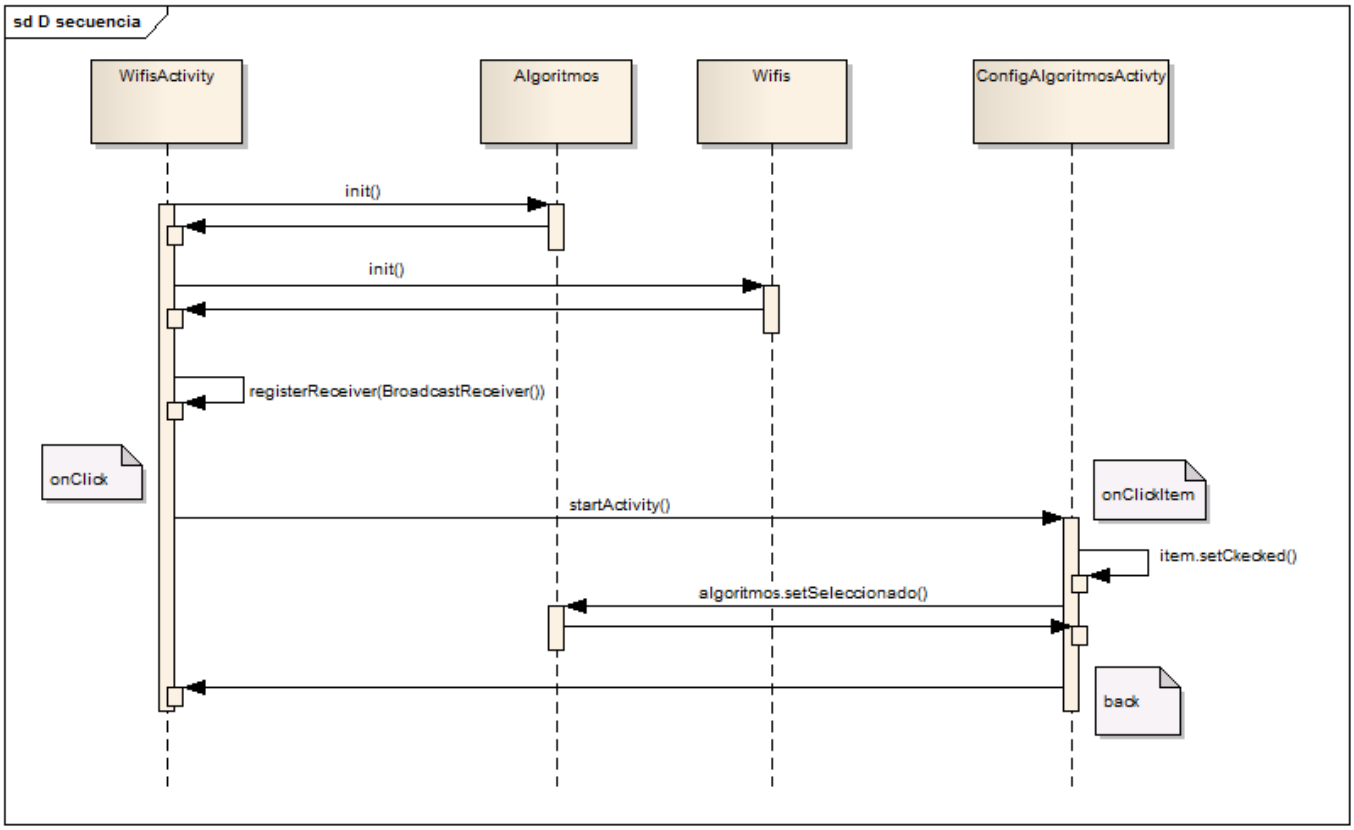

Figura 44: Cambio en la configuración de los algoritmos seleccionados

- Ejecución completa: La secuencia de llamadas entre clases para realizar una ejecución completa, desde la pantalla principal hasta que se muestran los resultados de precisión de cada algoritmo.

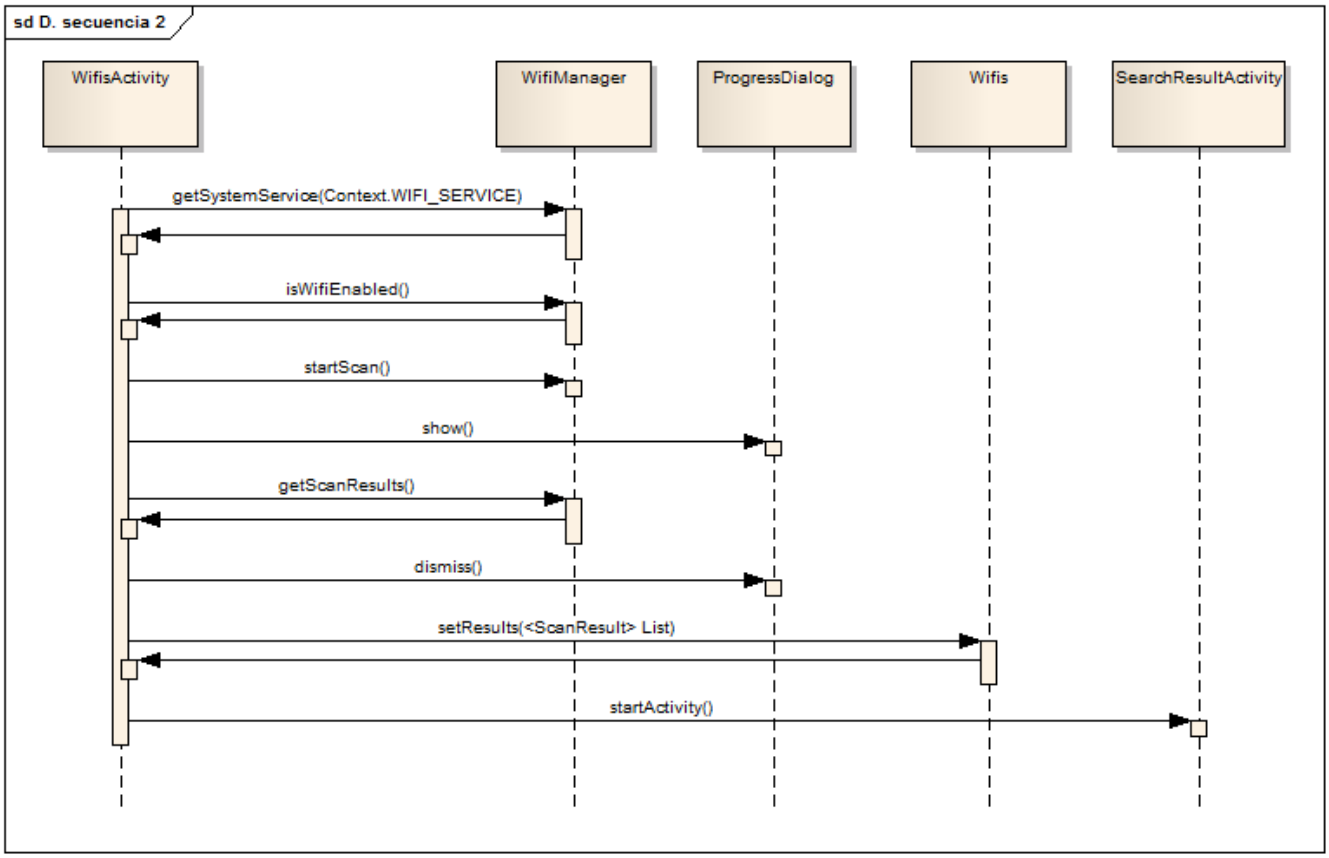

Figura 45: Ejecución completa (Parte 1)

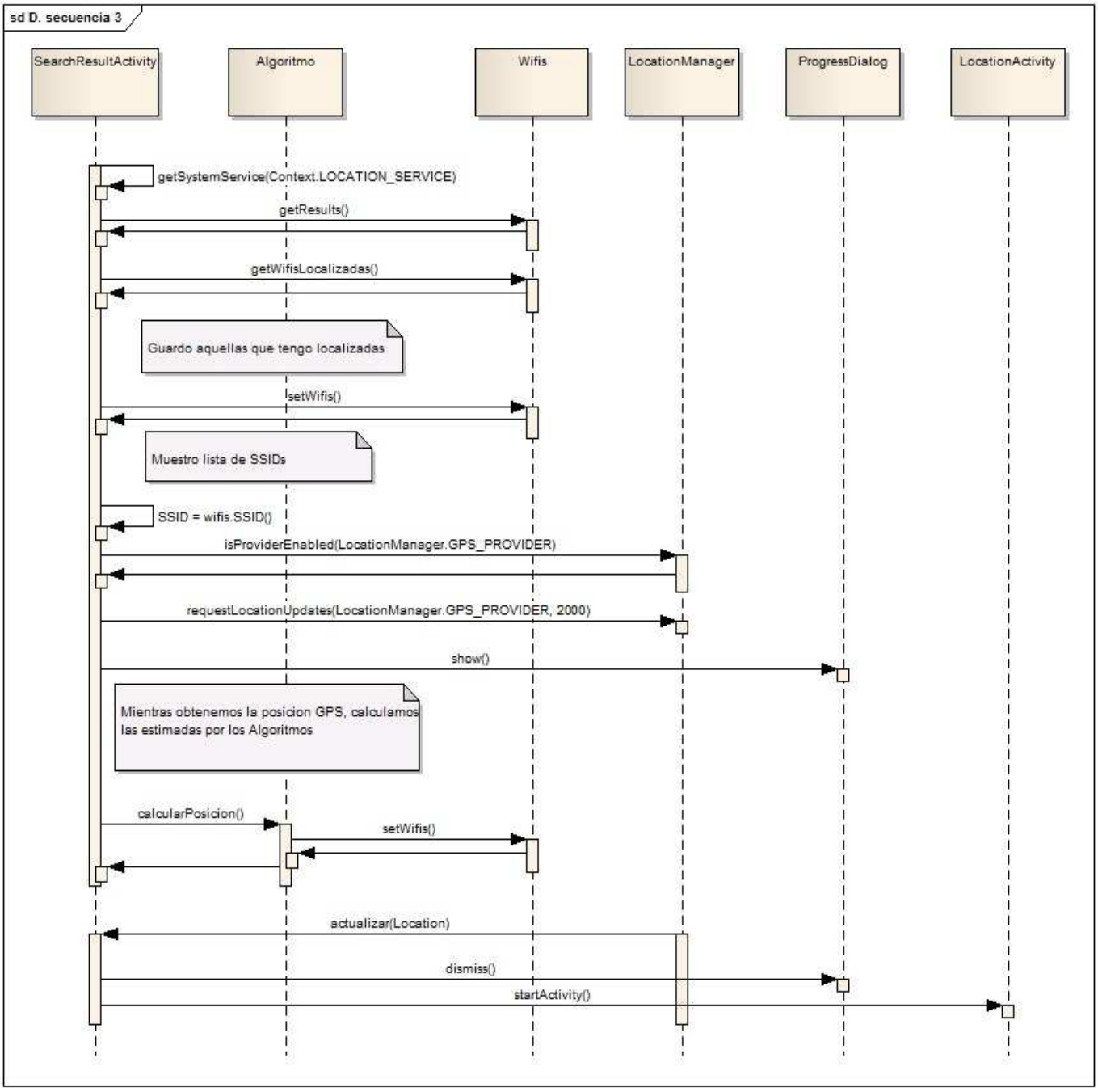

Figura 46: Ejecución completa (Parte 2)

6.2.10 Implementación

La implementación de esta aplicación requiere más complejidad en el desarrollo, ya que utiliza más elementos gráficos: listas personalizadas, botones, mensajes de progreso y más pantallas entre las que navegar. Además de implementarla para llevar a cabo las pruebas de los

diferentes algoritmos, permitirá adquirir los conocimientos necesarios sobre la implementación haciendo uso del GPS y de las redes Wifi, para implementarlo en la aplicación final.

Los algoritmos finalmente implementados para experimentación son los siguientes:

Nombre	Descripción/Operaciones
Algoritmo 1	Transforma los valores de RSSI a unidades naturales, normaliza y usa las raíces de las potencias normalizadas como pesos.
Algoritmo 2	Transforma los valores de RSSI a unidades naturales, normaliza y usa las potencias normalizadas como pesos.
Algoritmo 3	Transforma los valores de RSSI a unidades naturales, normaliza, descarta los datos atípicos y usa las raíces de las potencias normalizadas como pesos.
Algoritmo 4	Normaliza los valores de RSSI en unidades logarítmicas y usa las potencias normalizadas como pesos.
Algoritmo 5	Normaliza los valores de RSSI en unidades logarítmicas, descarta los datos atípicos y usa las potencias normalizadas como pesos.
Algoritmo 6	Normaliza los valores de RSSI en unidades logarítmicas y usa las raíces de las potencias normalizadas como pesos.
Algoritmo 7	Normaliza los valores de RSSI en unidades logarítmicas, descarta los datos atípicos y usa las raíces de las potencias normalizadas como pesos.

Tabla 6: Algoritmos implementados

6.2.10.1 Redes WiFi en Android

Se verán en este subapartado algunos aspectos relacionados con las posibilidades que ofrece Android para obtener la información de las distintas señales WiFi y de los APs emisores.

Android dispone en su API de un conjunto de clases para acceder a la información recibida a través de la tarjeta de red inalámbrica. Se muestran a continuación las que resultan más relevantes para el diseño del sistema de posicionamiento.

La clase WifiManager [25] proporciona de cada red WIFI detectada, un objeto WifiInfo [26], que contiene toda la información asociada a la red, y del que se puede obtener el parámetro de la potencia recibida o RSSI.

Aunque no se menciona directamente, a través de otros parámetros relacionados con el RSSI, se puede deducir que se obtienen los valores en unidades logarítmicas, por tanto, para algunos algoritmos se tendrán que convertir estos valores a unidades naturales.

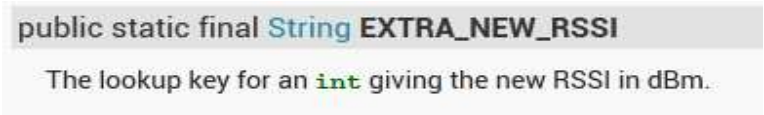

Figura 47: Constante EXTRA_NEW_RSSI

La clase WifiManager también proporciona métodos que permiten calcular y comparar niveles de señal con los métodos:

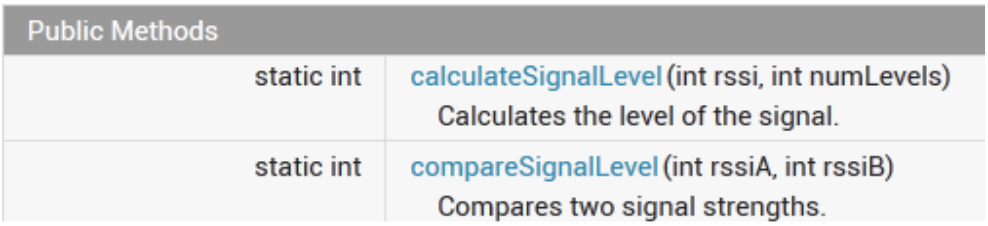

Figura 48: Métodos WifiManager

A través de los métodos anteriores se podrán obtener los valores de RSSI y a partir de estos valores y conocidas las posiciones de los APs, operar matemáticamente para obtener la posición del usuario, siguiendo los algoritmos definidos anteriormente y sus variaciones.

6.2.10.2 Pruebas verificadas

Tras la implementación se verifica el conjunto de pruebas, los resultados obtenidos son:

Prueba	Estado
P-1	Verificada
P-2	Verificada
P-3	Verificada
P-4	Verificada
P-5	Verificada
P-6	Verificada
P-7	Verificada
P-8	Verificada

Tabla 7: PosicionWiFi Pruebas Verificadas

Capítulo 7: Experimentación y Resultados

Las aplicaciones descritas en el apartado anterior se utilizarán con el fin de obtener resultados comparativos de la precisión de los distintos algoritmos. Finalmente se elegirá el que mejores resultados proporciona para implementarlo en el sistema. El resultado proviene de la ejecución de repetidas pruebas en diferentes escenarios.

7.1 Desarrollo de las pruebas

Las pruebas se desarrollarán espacio abierto, ya que requerimos conectividad GPS en la ejecución de las pruebas, para poder obtener el margen de desviación con respecto a la posición GPS real.

El procedimiento para el desarrollo de las pruebas se describe a continuación:

1. Colocación de acuerdo a la topología de la prueba de los puntos de acceso.
2. Obtención por medio de la aplicación *Posición GPS* de las posiciones GPS de cada punto de acceso.
3. Incorporación de las posiciones obtenidas al fichero de configuración de la aplicación *Posición Wifi*.
4. Ejecución de la aplicación Posición Wifi, con los algoritmos que queramos comparar y obtención de la desviación para cada algoritmo.
5. Anotación de los resultados para posterior comparación.

Para cada topología o prueba, realizaremos 3 medidas diferentes, variando el punto de medida desde el que se ejecuta la aplicación Posición Wifi, de esta forma los resultados obtenidos serán más fiables.

El recinto en el que se realizan las pruebas es de forma cuadrada y de aproximadamente 150m2. El terminal utilizado como cliente es el *smartphone* Samsung Galaxy S [27]. Inicialmente se realizaron las pruebas utilizando *smartphones* Samsung Galaxy S en modo AP, pero se comprobó que las potencias emitidas eran bajas, insuficientes para completar todas las pruebas. Consecuentemente se decidió utilizar como APs los dispositivos de la marca Zyxel, modelo P660HWP-D1 [28].

7.2 Pruebas realizadas

A continuación se describe cada prueba realizada, se representa su configuración y los resultados obtenidos.

7.2.1 Prueba 0

La prueba inicial consiste en utilizar solamente un AP, pero la posición estimada sería la posición del propio AP, y la diferencia con la posición sería la distancia de separación del AP. Esta prueba no aporta ningún dato relevante al estudio.

7.2.2 Prueba 1

La primera prueba es la más sencilla, sólo se emplean 2 APs y se coloca el dispositivo móvil en un punto intermedio de la línea que los une. Para cada medición se varía la posición acercándose o alejándose de uno de los APs.

Figura 49: Topología Prueba 1

Resultados en metros respecto a la posición GPS:

Algoritmo	Medición 1	Medición 2	Medición 3
Algoritmo 1	1.79	1.77	1.80
Algoritmo 2	1.89	1.95	1.92
Algoritmo 3	1.79	1.77	1.80

Algoritmo 4	2.77	3.13	3.02
Algoritmo 5	2.77	3.13	3.02
Algoritmo 6	3.19	3.06	3.16
Algoritmo 7	3.19	3.06	3.16

Prueba 13: Experimentación Prueba 1

La posición resultante se estima en la recta que une ambos APs, más cerca del que se recibe con mayor potencia. La precisión varía en función del algoritmo y algunos algoritmos arrojan exactamente los mismos valores, esto se debe a que no se descarta ningún AP para los cálculos; pero los resultados obtenidos son buenos. Esto se debe a que es un escenario muy particular dado que el dispositivo móvil se sitúa en la recta que une los 2 APs. Generalmente no se dará esta situación, por tanto estos resultados dan una primera idea, pero no son muy representativos de una situación real.

7.2.3 Prueba 2

Para la segunda prueba también se utilizaran 2 APs, pero esta vez se colocará el dispositivo móvil fuera de la línea imaginaría que los une.

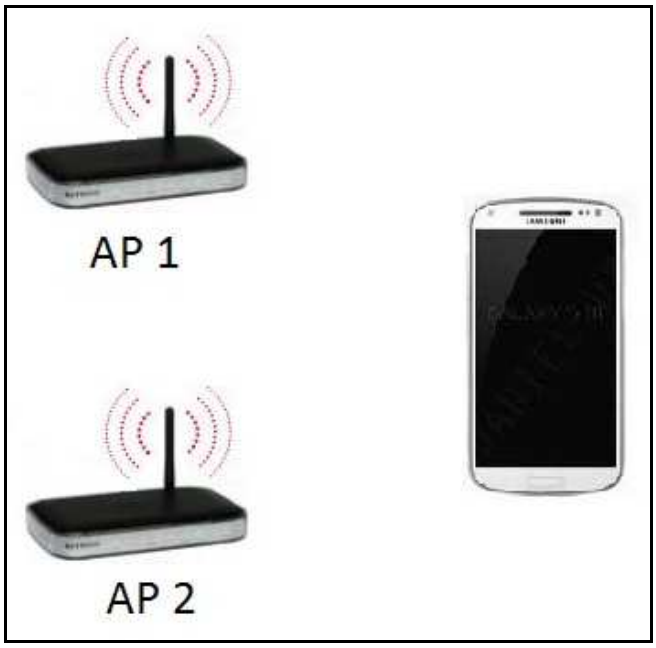

Figura 50: Topología Prueba 2

Resultados en metros respecto a la posición GPS:

Algoritmo	Medición 1	Medición 2	Medición 3
Algoritmo 1	6.14	6.33	6.37
Algoritmo 2	6.92	7.15	7.01
Algoritmo 3	6.14	6.33	6.37
Algoritmo 4	6.38	6.60	6.56
Algoritmo 5	6.38	6.60	6.56
Algoritmo 6	6.54	6.59	6.73
Algoritmo 7	6.54	6.59	6.73

Prueba 14: Experimentación Prueba 2

Al solo tener 2 APs, la posición resultante se estima en la recta que une ambos APs, más cerca de aquel que se recibe con mayor potencia. Al estar alejado el dispositivo móvil de esta recta, los resultados obtenidos son peores que en la prueba anterior, pero pueden ser más representativos de un caso real.

7.2.4 Prueba 3

En esta tercera prueba se introduce un AP más, el objetivo es mejorar los resultados al disponerse de más información. El dispositivo móvil se situará dentro del triángulo que forman los puntos de acceso.

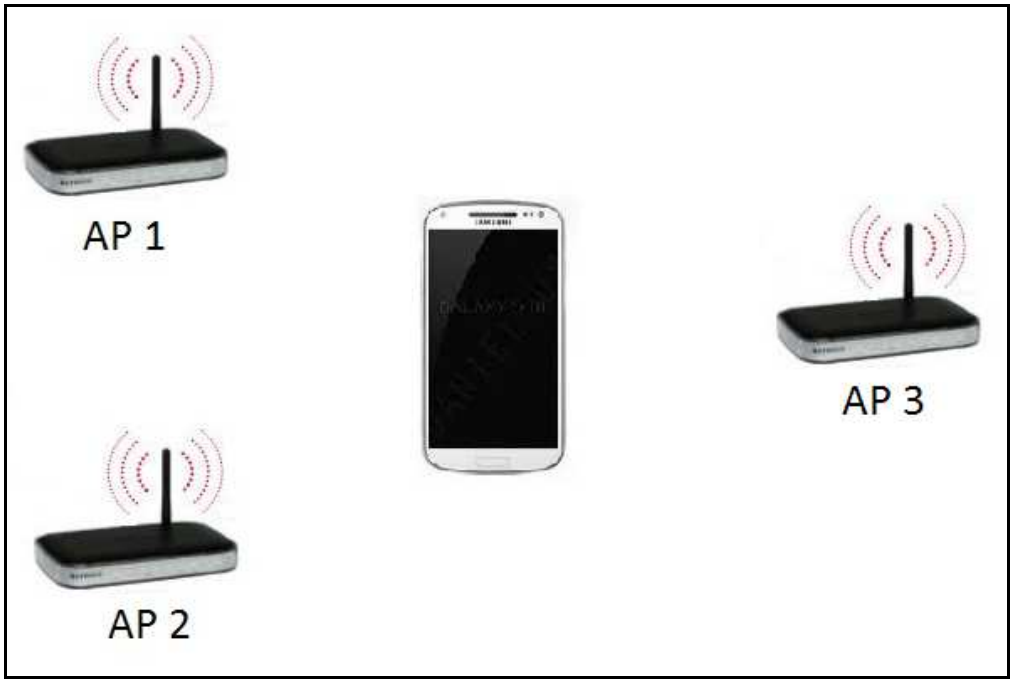

Figura 51: Topología Prueba 3

Resultados en metros respecto a la posición GPS:

Algoritmo	Medición 1	Medición 2	Medición 3
Algoritmo 1	2.75	2.78	2.83
Algoritmo 2	3.57	3.70	3.63
Algoritmo 3	2.75	2.78	2.83
Algoritmo 4	5.25	5.27	5.34
Algoritmo 5	5.25	5.27	5.34
Algoritmo 6	5.12	5.19	5.13
Algoritmo 7	5.12	5.19	5.13

Prueba 15: Experimentación Prueba 3

Esta tercera prueba se desarrolla en un escenario más realista. La posición calculada cae en algún punto dentro del triángulo que forman los APs, pero a cierta distancia de la posición obtenida por el GPS. Tampoco los algoritmos que realizan descartes, están desechando la información de alguno de los APs, por lo que sigue habiendo algoritmos con resultados idénticos. Y al operar en unidades naturales la posición calculada se ajusta más a la real.

7.2.5 Prueba 4

Este escenario consta, como el anterior, de 3 puntos de acceso, pero dispuestos en línea y situando el dispositivo móvil fuera de la línea que los une.

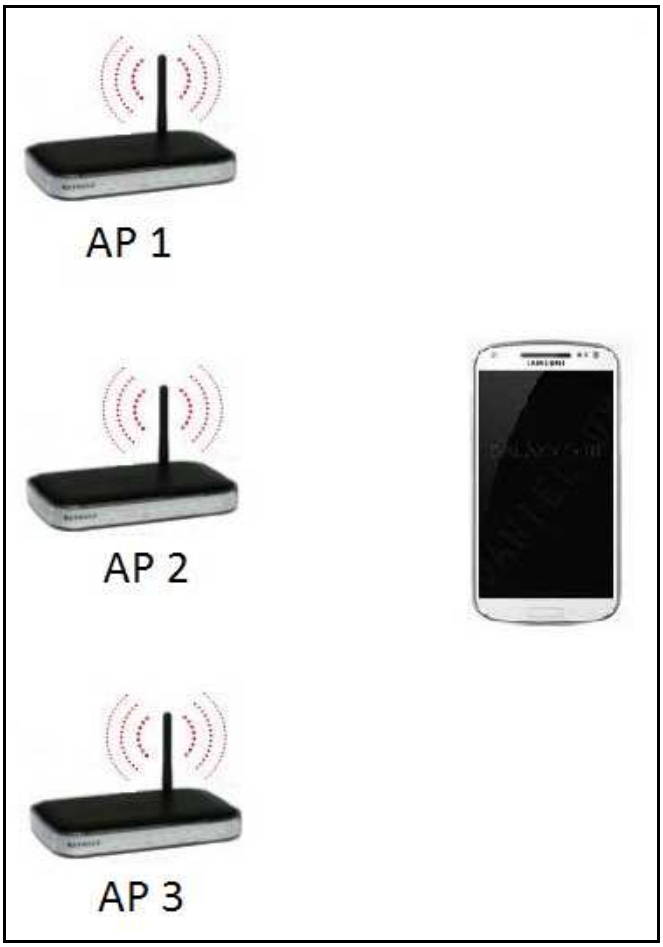

Figura 52: Topología Prueba 4

Resultados en metros respecto a la posición GPS:

Algoritmo	Medición 1	Medición 2	Medición 3
Algoritmo 1	5.34	5.31	6.27
Algoritmo 2	6.92	7.15	7.01
Algoritmo 3	5.34	5.31	6.27
Algoritmo 4	6.46	6.80	6.68

Algoritmo 5	6.46	6.85	6.68
Algoritmo 6	6.62	6.78	6.47
Algoritmo 7	6.62	6.78	6.47

Prueba 16: Experimentación Prueba 4

Al estar situados los APs sobre la misma recta, la posición estimada caerá sobra la propia recta. Como el dispositivo móvil se encuentra alejado de esta línea, los resultados se alejan del obtenido por GPS aproximadamente la distancia que separa al dispositivo de esa línea imaginaria. En general todos los algoritmos ofrecen peores y de precisiones parecidas.

7.2.6 Prueba 5

Se añade un nuevo AP y se sitúa el *smartphone* dentro del área encerrada por los APs.

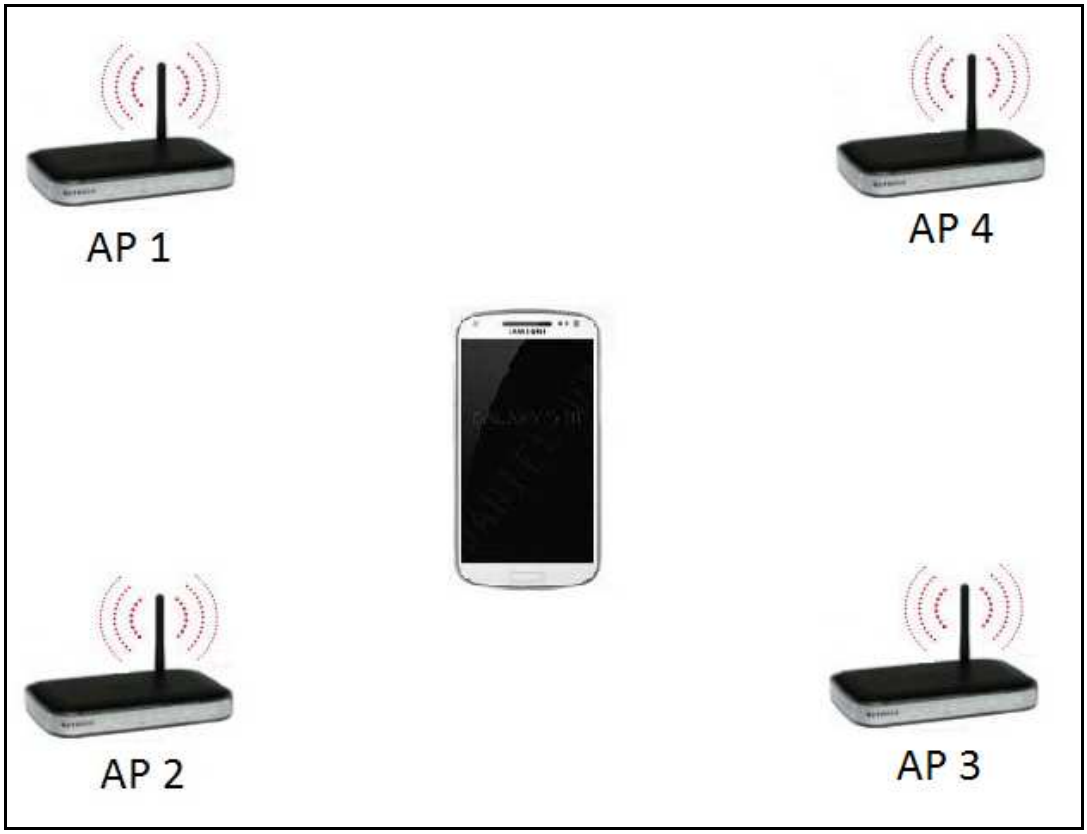

Figura 53: Topología Prueba 5

Resultados en metros respecto a la posición GPS:

Algoritmo	Medición 1	Medición 2	Medición 3
Algoritmo 1	2.62	2.52	2.60
Algoritmo 2	2.95	2.83	2.91
Algoritmo 3	2.99	3.07	3.04
Algoritmo 4	5.82	5.76	5.72
Algoritmo 5	5.97	5.90	5.95
Algoritmo 6	4.94	4.99	5.11
Algoritmo 7	5.04	5.17	5.19

Prueba 17: Experimentación Prueba 5

Al tener mayor número de APs de referencia, se dispone de más información. En este escenario los algoritmos que implementan la variación de desechar datos si lo hacen, por tanto ya no se recogen resultados duplicados. Lo que se observa es que los algoritmos que realizan los descartes muestran peores resultados. Y los que mejores resultados presentan son los que trabajan con unidades naurales.

7.2.7 Prueba 6

Para el último escenario de pruebas añadimos un nuevo AP y se sitúa el dispositivo móvil dentro del recinto limitado por los APs. 5 es un número suficiente para realizar pruebas, no añadiremos más, pues en situaciones reales es improbable encontrar escenarios con mayor número de puntos de acceso.

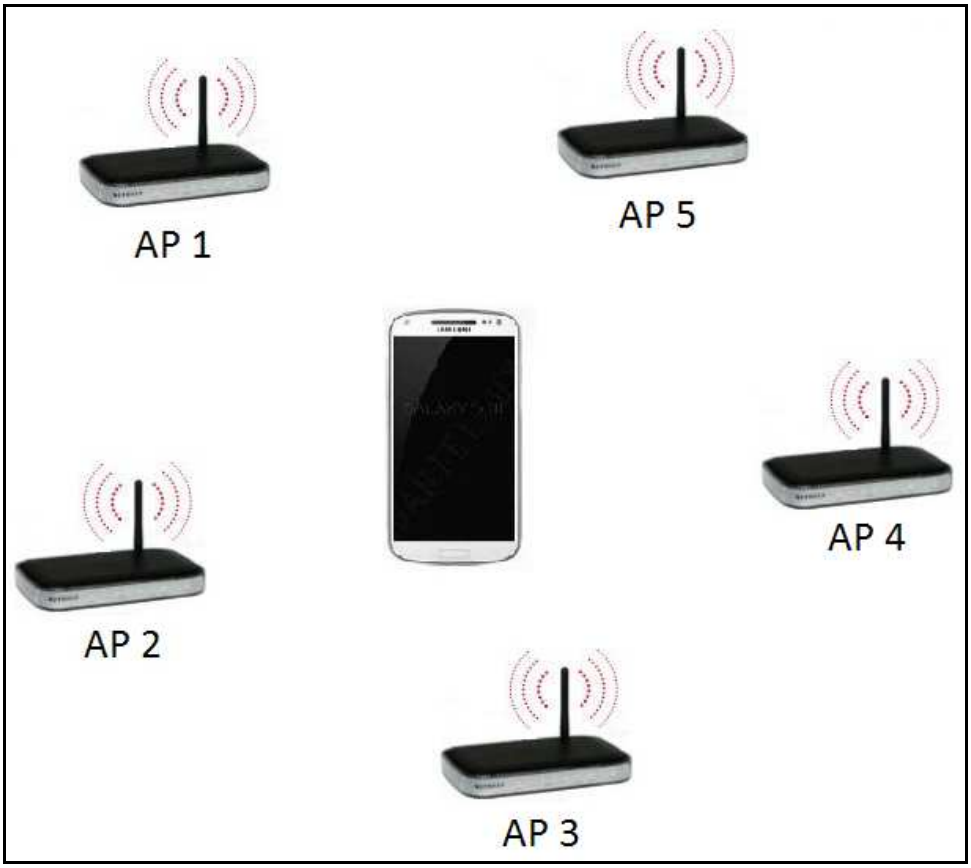

Figura 54: Topología Prueba 6

Resultados en metros respecto a la posición GPS:

Algoritmo	Medición 1	Medición 2	Medición 3
Algoritmo 1	2.34	2.38	2.46
Algoritmo 2	3.22	3.17	3.19
Algoritmo 3	2.92	2.86	2.88
Algoritmo 4	4.83	4.69	4.77
Algoritmo 5	5.11	5.06	5.09
Algoritmo 6	4.98	4.93	4.88
Algoritmo 7	5.25	5.29	5.20

Prueba 18: Experimentación Prueba 6

Al igual que en la prueba anterior, se aumenta la información disponible y se aprecia una mejora en los resultados, aunque no muy grande. De nuevo los algoritmos que descartan información calculan con menos precisión y los que mejores resultados presentan son los que calculan con unidades naturales.

7.3 Discusión de Resultados

A través de las diferentes pruebas, con escenarios distintos, algunos muy particulares, se ha podido ver la respuesta ofrecida por los diferentes algoritmos, comparada con la información obtenida por el sensor GPS del dispositivo móvil.

Por la naturaleza de los algoritmos, en los casos con menos información de referencia (2-3 APs) los resultados son muy dependientes de la colocación de los APs. Al estar colocados en línea, se aprecia que cuando el dispositivo se aleja, los resultados empeoran considerablemente.

Según se va aumentando la información de referencia, es decir, el número de puntos de acceso, los resultados van mejorando, aunque hasta cierto punto (5 APs), donde la mejora no es tan notable. En el caso de los algoritmos que descartan datos atípicos, esto no sucede, ya que al haber mayor número de APs, realizan descartes y perdiendo esta información, los resultados obtenidos son menos precisos.

Hay que tomar estos resultados de forma crítica, ya que las pruebas se han realizado en un entorno controlado, donde las distancias no son grandes y donde no se sufren efectos de interferencias o reflexiones y siendo las condiciones atmosféricas favorables.

Pero las conclusiones obtenidas son que los resultados son fuertemente dependientes de la colocación de los APs o de la posición del dispositivo móvil respecto de los APs y que mejoran cuanto mayor es el número de APs disponibles para hacer los cálculos. Eliminar información no resulta una estrategia ventajosa, dado que el número de puntos de referencia no es muy elevado. De la misma forma, operar con unidades logarítmicas no mejora los resultados, la no linealidad de algunas operaciones produce mayor nivel de error en los cálculos.

7.4 Elección del algoritmo

Tras analizar los resultados de las diferentes pruebas se concluye que el algoritmo que mejores resultados arroja es el Algoritmo 1. En situaciones desfavorables los resultados son tan imprecisos como en el resto de algoritmos, pero con una colocación adecuada de los APs, la precisión media que ofrece es de 2,5 metros.

Esta precisión es referida a la posición que obtendríamos con el GPS del *smartphone*, la cual tiene también un margen de error de unos 3 metros [29]. Por tanto el error acumulado resulta de 5,5 metros.

El margen de error obtenido es razonable y hace que los resultados sean suficientemente buenos para el sistema que se diseña, ya que no se pretenden rutas precisas, sino informar a un usuario de la posición de sus contactos. Y un error en esta posición de 5,5 metros no es elevado para los lugares en que se utilizará la aplicación, por tanto los resultados serán fiables.

Será entonces este algoritmo el que se implementará en la aplicación cliente par a calcular la posición del usuario en entornos en los que no sea posible obtenerla por medio del GPS.

Capítulo 8: Gestión del Proyecto

En este capítulo se tratará la gestión del proyecto. Cómo se ha planificado, los medios utilizados, la gestión de versiones del software y el presupuesto necesario.

8.1 Planificación

En este apartado se detallan las tareas principales que componen el proyecto, representando la duración necesaria para cada una de ellas en un diagrama de Gantt.

8.1.1 Principales tareas

- Análisis de Negocio:
  - Toma de Requisitos: Recopilación de los requisitos para la aplicación.
  - Procesos: Definición de los procesos internos del sistema.
  - Análisis tecnológico: Estudio de las tecnologías que hacen viable el sistema.
  - Arquitectura: Diseño de la arquitectura del sistema completo y de sus componentes.

- Sistema de Posicionamiento:
  - Diseño del sistema: Estudio y diseño de un sistema de posicionamiento basado en APs.
  - Análisis Apps: Análisis de las aplicaciones necesarias para estudio y prueba del sistema de posicionamiento.
  - Diseño Apps: Diseño de las aplicaciones.
  - Implementación: Programación de las aplicaciones de estudio.
  - Pruebas: Verificación del funcionamiento de las aplicaciones y solución de posibles errores.
  - Experimentación: Uso de las anteriores aplicaciones para obtener datos relativos al sistema y análisis de esos datos.

8.1.1 Planificación inicial

Se muestran a continuación las tareas y plazos para cada tarea estimados inicialmente. Como se verá posteriormente el desarrollo real del proyecto superó los plazos iniciales, si bien las tareas necesarias son las mismas.

A priori se consideraron casi 6 meses de trabajo, disponiendo únicamente de 4 horas diarias para este proyecto. El tiempo total estimado es de 115 días efectivos repartidos como se muestra en la Figura 55, lo que supone unas 460 horas.

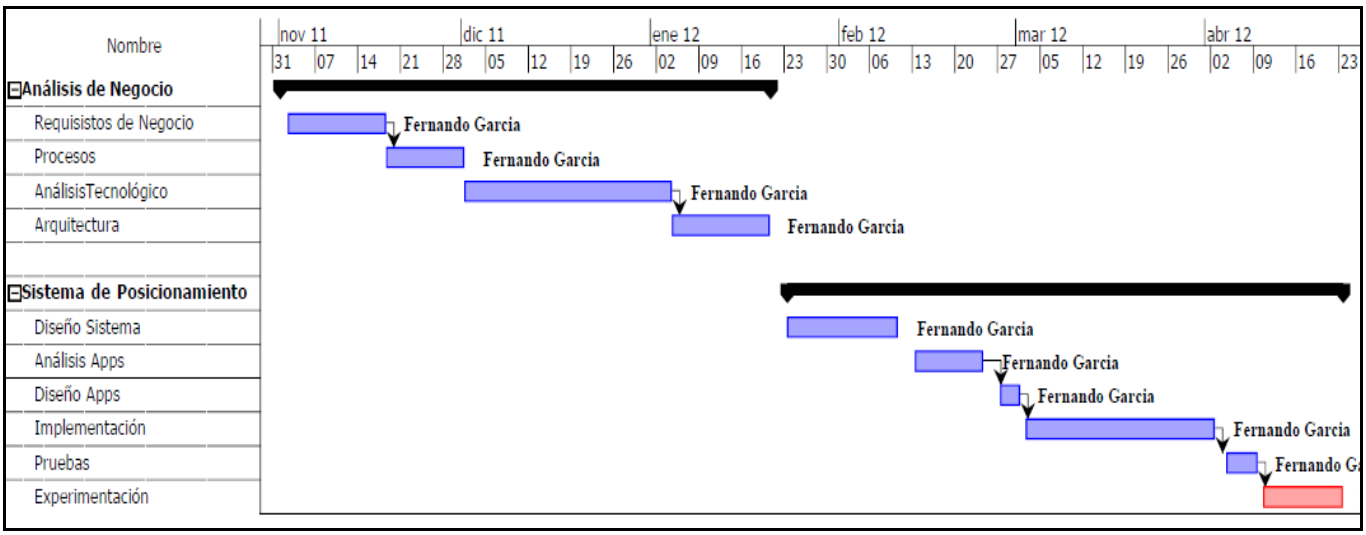

Figura 55: Planificación inicial

8.1.2 Planificación real

Se muestra en este apartado los plazos reales requeridos en cada una de las tareas. El tiempo total necesitado para el proyecto supera los 6 meses considerados inicialmente, sin embargo la desviación no es muy grande.

Finalmente se necesitaron para completar todas las tareas del proyecto 130 días efectivos de trabajo repartidos como se muestra en la Figura 56, unas 520 horas, lo que supone 60 horas adicionales empleadas en el proyecto.

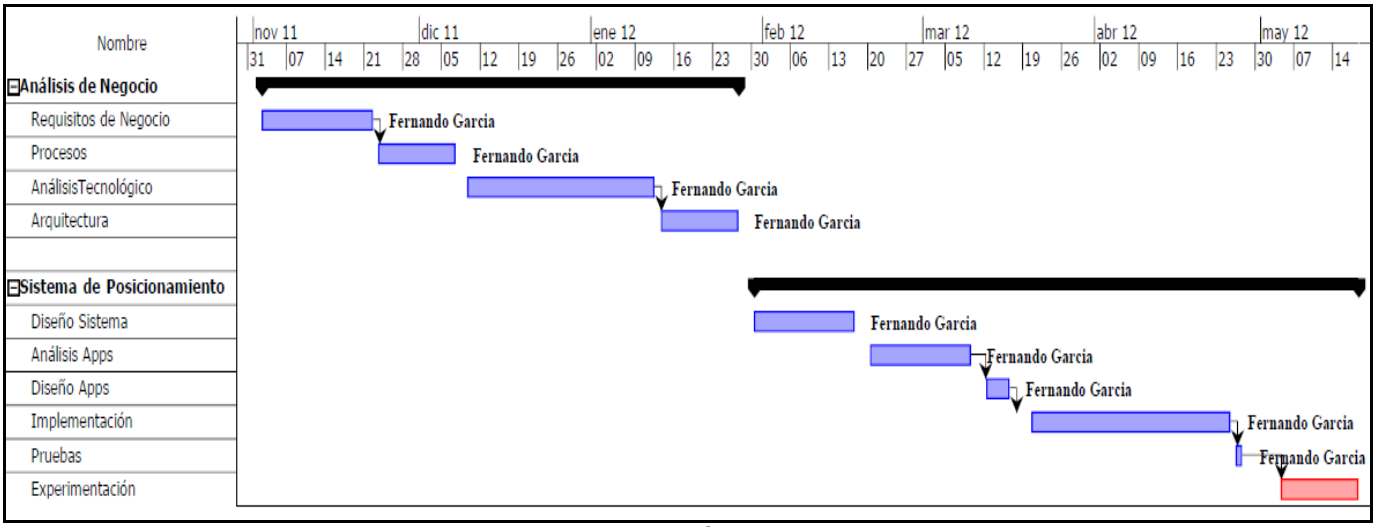

Figura 56: Planificación real

8.2 Control de versiones

El control de versiones es una cuestión muy importante a tener en cuenta en los proyectos de software. Consiste en el manejo de los cambios en la información, especialmente en el código fuente de las aplicaciones. Las herramientas de control de versiones son críticas para los programadores, quienes hacen y deshacen continuos cambios en el software.

El control de versiones es necesario en cualquier proyecto, no solo para para tener un historial de los cambios y versiones anteriores, sino como utilidad para compartir el software y las modificaciones cuando intervienen varios desarrolladores.

Existen diferentes productos para el control de versiones, pero el que se ha utilizado en este proyecto se llama Subversion [30]. La experiencia de los desarrolladores del proyecto con este sistema, su sencillo funcionamiento y la existencia de repositorios privados gratuitos justifican su elección.

8.2.1 SVN

Subversion (SVN) es un sistema de control de versiones gratuito y de código fuente abierto. Gestiona ficheros y directorios de forma temporal. Hay un árbol de ficheros en un repositorio central, que es un servidor de ficheros ordinario, excepto porque recuerda todos los cambios hechos a sus ficheros y directorios. Esto le permite recuperar versiones antiguas de sus datos y examinar el historial de cambios.

Subversion proporciona:

- Versionado de directorios: que sigue los cambios sobre árboles de directorios completos a través del tiempo.
- Historial de versiones: se puede añadir, borrar, copiar, y renombrar ficheros y directorios. Y cada fichero nuevo añadido comienza con un historial nuevo.
- Versionado de metadatos: cada fichero y directorio tiene un conjunto de propiedades asociado. Las propiedades son versionadas a través del tiempo, al igual que el contenido de los ficheros.
- Ramificación y etiquetado eficientes: Subversion crea ramas y etiquetas simplemente copiando el proyecto.

8.2.2 Servidor SVN

Para alojar las diferentes versiones del software, existen empresas que ofrecen servidores de control de versiones, en este caso de Subversion, de forma gratuita pero con ciertas limitaciones en el caso de proyectos privados.

Entre las diferentes posibilidades, se decidió utilizar XP-DEV [31] por cumplir los requisitos necesarios para el volumen de datos y número de usuarios de este proyecto.

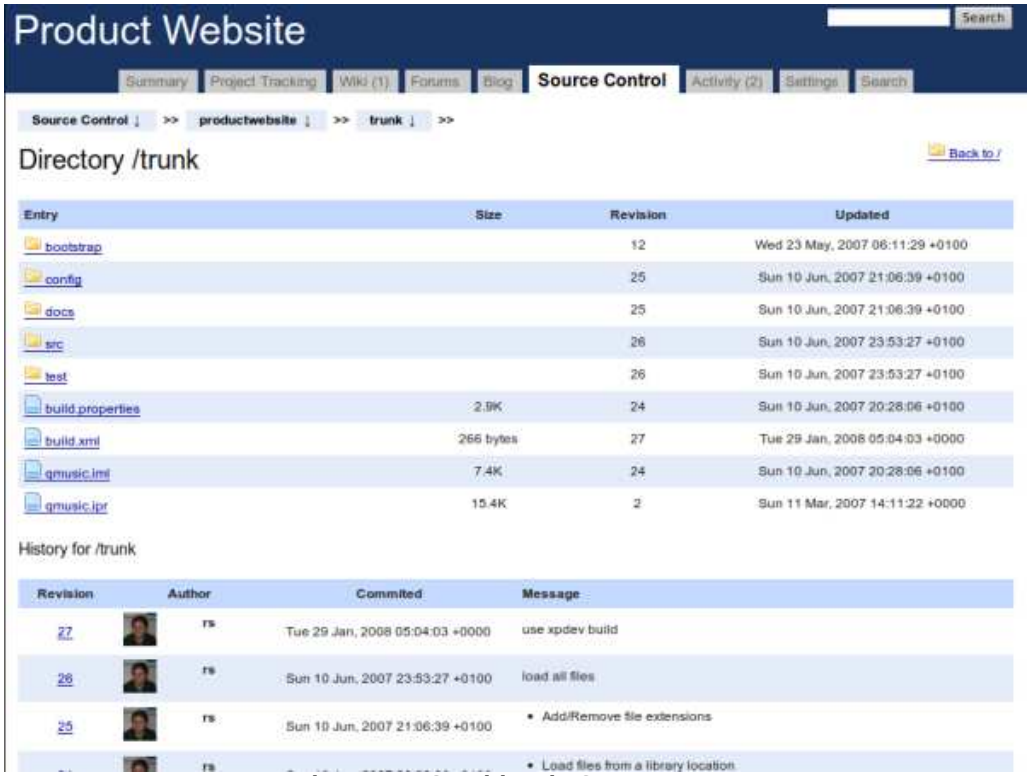

Figura 57: Servidor de SVN XP-DEV

Algunas de las características que ofrece este servicio son:

- Fácil configuración
- Emails informando de los cambios realizados
- Explorador web de los repositorios
- Visualización de ficheros y diferencias entre versiones.
- Seguimiento de bugs
- Soporte para múltiples usuarios
- Limitación a 10MB de espacio y 2 proyectos

Y de la infraestructura de servidores:

- Datacenters rápidos y seguros, con soporte 24x7x365
- Datos almacenados en servidores dedicados con discos duros encriptados.
- Backups redundantes en datacenters independientes

8.2.3 Subclipse

Como cliente del servicio SVN para introducir los cambios en el servidor, recuperar versiones, ver cambios de código, etc, se ha utilizado un plugin para Eclipse llamado Subclipse [32].

Subclipse es un cliente para Subversion, que se integra en el IDE Eclipse lo que permite realizar los cambios y recuperar versiones de forma sencilla, desde el propio IDE.

Las principales operaciones que permite son:

- import: incorporar un proyecto inicial al servidor en el que se van a guardar las distintas versiones. Esta operación se realiza sólo una vez.
- checkout: descargar una copia del proyecto a una carpeta local para trabajar sobre la misma (copia de trabajo). Cada uno de los desarrolladores realizará esta operación cuando empieza a colaborar en un proyecto.
- update: descarga la última versión de cada uno de los ficheros que forman parte del proyecto para actualizar la copia de trabajo local.
- add, delete, copy, move: Se pueden añadir, eliminar, copiar o mover ficheros al servidor de control de versiones.
- commit: enviar los cambios realizados al servidor.

8.3 Medios técnicos

Los medios técnicos y herramientas empleadas para la realización del proyecto, tanto hardware como software, se presentan a continuación:

8.3.1 Dispositivos Hardware

En la Tabla 8 se muestran los dispositivos de hardware utilizados:

Tipo	Nombre	Unidades
PC	Asus A52F K52F	1
Dispositivo móvil	Samsung Galaxy S [27]	1
AP	Zyxel p660HWP-D1 [28]	5

Tabla 8: Hardware

8.3.2 Aplicaciones Software

La lista de programas de software empleados en el desarrollo del proyecto se recoge en la Tabla 9:

Tipo	Nombre	Página Web
Sistema operativo	Windows 7 Home Premium	http://emea.microsoftstore.com/es/es-ES/Microsoft/Windows-7-Home-Premium
IDE	Eclipse Indigo	http://www.eclipse.org/indigo
Pugin Android - Eclipse	ADT Plugin	http://developer.android.com/tools/sdk/eclipse-adt.html
Servidor	Google App Engine	https://appengine.google.com
Servidor Notificaciones	C2DM	https://developers.google.com/android/c2dm
Plugin App Engine - Eclipse	Google Plugin for Eclipse	https://developers.google.com/eclipse
Servidor Subversion	xp-dev	http://xp-dev.com
Plugin Subversion - Eclipse	Subclipse	http://subclipse.tigris.org
Ofimática	Microsoft Office 2010	http://office.microsoft.com/es-es/products
Gestión de Proyectos	OpenProject	http://sourceforge.net/projects/openproj
Diagramas	DIA	https://live.gnome.org/Dia

Tabla 9: Software

8.4 Presupuesto

Se estiman los costes totales de este proyecto, desglosando los conceptos en costes de personal, costes de materiales: software y hardware y costes indirectos. Al resultado de coste total, se añadirá un porcentaje de beneficio que se obtendrá a la hora de vender el proyecto.

8.4.1 Costes de Personal

Los costes de personal se refieren únicamente a los honorarios del Ingeniero de Telecomunicaciones encargado del desarrollo del proyecto, con una carga de trabajo de 520 horas, obtenida de la planificación real del proyecto.

Concepto	Horas	Honorarios	Total
Ingeniero de Telecomunicaciones	520 h.	18€/h	9360,00 €
TOTAL			**9360,00 €**

Tabla 10: Costes Personal

8.4.2 Costes de Hardware

Se detallan el hardware utilizado junto con los precios, para calcular los costes asociados a la realización del proyecto.

Concepto	Precio unitario	Vida útil	Tiempo de uso	Unidades	Coste
Asus A52F K52F	600€	36 meses	6 meses	1	100,00 €
Samsung Galaxy S	264€	24 meses	6 meses	1	66,00 €
Zyxel p660HWP-D1	90€	48 meses	6 meses	5	56,25 €
TOTAL					**222,25 €**

Tabla 11: Costes Hardware

8.4.3 Costes de Software

Para este proyecto no se ha incurrido en costes de software. Todos los programas utilizados son gratuitos y de código libre. A excepción del sistema operativo del PC y la herramienta ofimática, incluidos con el PC, cuyos costes están incluidos en los costes del PC y que podrían ser sustituidos por aplicaciones similares de código libre y gratuitas.

8.4.4 Costes Indirectos

Existen una serie de servicios que repercuten unos costes indirectos en el proyecto, son básicamente la luz y el acceso a Internet.

Concepto	Precio	Tiempo de uso	Coste
Luz	45 €/mes	6 meses	270,00 €
Internet	40 €/mes	6 meses	240,00 €
TOTAL			**510,00 €**

Tabla 12: Costes indirectos

8.4.5 Costes Totales

Se muestra en la siguiente tabla la suma de los costes anteriores, lo que resulta en el coste total en el que se ha incurrido durante la realización de este proyecto.

Concepto	Coste
Costes de Personal	9360,00 €
Costes de Hardware	222,25 €
Costes Indirectos	510,00 €
Total antes de IVA	**10092,25 €**
IVA 18%	1816,60 €
TOTAL	**11908,85 €**

Tabla 13: Costes totales

8.4.6 Precio Final

El precio final de venta del proyecto resulta de añadir un 30% de beneficio a los costes totales del proyecto, quedando de la siguiente manera:

Concepto	Coste
Costes totales	10092,25 €
IVA 18%	1816,60 €
SUBTOTAL	**11908,85 €**
30% beneficio	3572,65 €
TOTAL	**15481,50 €**

Tabla 14: Precio Final

Capítulo 9: Conclusiones

En este capítulo se resumirán las tareas realizadas y como han conducido a alcanzar los objetivos planteados inicialmente y se propondrán también unas líneas de trabajo futuras como continuación del proyecto.

9.1 Conclusiones

A lo largo de este proyecto se han combinado fases de investigación, de diseño y de desarrollo, logrando cumplir con los objetivos básicos que se planteaban inicialmente y ofreciendo una visión completa de los proyectos de esta índole.

Se realizó un estudio del sistema completo y un análisis de la aplicación a alto nivel, definiendo la arquitectura adecuada y comparando diferentes tecnologías para cada módulo, seleccionando las más ventajosas. Como se ha visto, este proyecto ha tratado con un gran conjunto de tecnologías y plataformas, servidores y servicios tan específicos como C2DM, con el objeto de ofrecer todas las características planteadas y con la complejidad que eso conlleva.

En este proyecto se han analizado y comparado sistemas y aplicaciones comerciales que desarrollaban funcionalidades de localización, en búsqueda de nuevas oportunidades. Se detectaron las carencias de los sistemas ya existentes, se analizaron las nuevas características y se implementaron soluciones para conseguir estas nuevas funcionalidades.

Uno de los principales objetivos consistió en crear un sistema de posicionamiento para lugares dónde otros sistemas convencionales como el GPS no podían funcionar, como en interiores. Se realizó el estudio y adaptación de técnicas de localización para interiores a los escenarios planteados. Se optó por adaptar la técnica de trilateración a partir de las medidas de potencia RSSI obtenidas de APs WiFi de referencia, introduciendo varios algoritmos para calcular la posición.

Se desarrolló de un conjunto de aplicaciones para experimentar con los diferentes algoritmos de cálculo de posiciones y que permitiera elegir el más óptimo. Se efectuaron pruebas de campo utilizando las aplicaciones en diferentes escenarios y analizando los resultados hasta conseguir un sistema que garantizara resultados fiables y suficientemente precisos.

La información sobre la localización de un usuario puede resultar muy valiosa, siempre que sea utilizada para los fines y por las personas que el usuario desea. La privacidad de esta información ha sido un aspecto considerado como muy importante desde el principio y ha estado presente en todas las cuestiones de diseño que pudieran comprometer esta privacidad.

Además de los objetivos alcanzados en el proyecto se han logrado otros personales. El desarrollo de aplicaciones para dispositivos móviles es un interés personal. Si bien partía de conocimientos avanzados de Java, este proyecto me ha dado la oportunidad de aprender a

programar para la plataforma Android, y participar en el desarrollo de una aplicación de cierta complejidad. El uso de un sistema de control de versiones ha resultado fundamental para compartir código y gestionar los cambios.

Finalmente cabe decir que este proyecto nació de una necesidad real, el hecho de ver materializado lo que tiempo atrás fue solo una idea, supone un gran éxito personal.

9.2 Trabajos futuros

Se definen las posibles líneas de trabajo futuras a tomar como continuación de este proyecto. Estas mejoras pasan por la adaptación a los cambios que se van produciendo en las distintas plataformas que utiliza el sistema, la mejora de los algoritmos de posicionamiento, la adaptación a otras plataformas móviles o el desarrollo de un API.

9.2.1 Adaptación

Las tecnologías sufren cambios continuamente y es necesario mantener las aplicaciones y adaptarlas a los cambios. Es el caso del servicio de notificaciones C2DM, que si bien sigue operando, va a ser migrado próximamente como se refleja en la propia web, ver Figura 58, a un nuevo servicio denominado *Google Cloud Messaging for Android (GCM)*. Será necesario estudiar los cambios y adaptar la aplicación a este nuevo servicio.

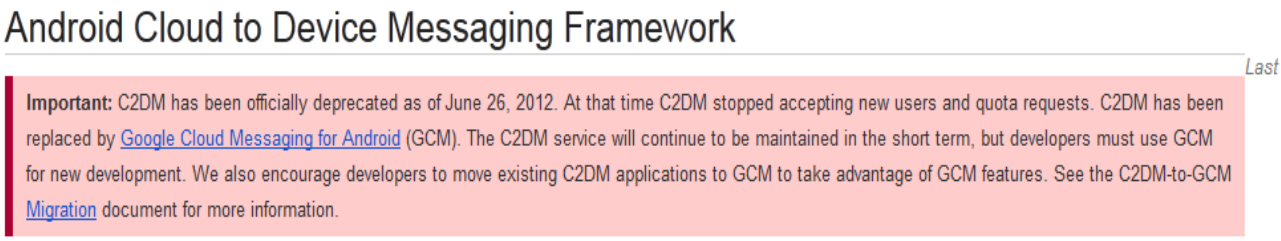

Figura 58: C2DM discontinuado

9.2.2 Mejora del algoritmo

El algoritmo seleccionado para el sistema de posicionamiento basado en APs ha ofrecido unos resultados válidos en cuanto a precisión. Pero este nivel de precisión es mejorable, además se puede seguir investigando algoritmos que den mejores resultados en entornos más ruidosos, con muchas reflexiones o interferencias.

9.2.3 Otras plataformas

El sistema se ha desarrollado sobre Android como sistema operativo de la parte cliente. Pero existen diferentes sistemas operativos alternativos, algunos con gran relevancia, como es el caso de iOS. No se pueden dejar de lado si se pretende crear un servicio que sea útil para

todos los potenciales usuarios. Sería interesante desarrollar clientes para estas plataformas y adaptar el sistema para que integrara a estos nuevos clientes.

9.2.4 API

El sistema implementado aporta interesantes funcionalidades, pero seguramente también sería muy útil para otras aplicaciones. Se podría desarrollar una librería accesible a otros desarrolladores a través de un API que permita a otras aplicaciones integrar en su lógica las funcionalidades que aporta el sistema desarrollado en este proyecto.

Capítulo 10: Acrónimos y Glosario

Se recogen en este capítulo los acrónimos y términos empleados en el documento, ordenados alfabéticamente y acompañados de su definición.

AP: Access Point. Un Punto de Acceso inalámbrico es un dispositivo que interconecta dispositivos de comunicación alámbrica para formar una red inalámbrica. El punto de acceso recibe la información, la almacena y la transmite entre la WLAN (Wireless LAN) y la LAN cableada.

API: Application Programming Interface. Es el conjunto de funciones y procedimientos, o métodos, en la programación orientada a objetos, que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción.

Contacto: Son aquellas personas o entidades de las que tenemos almacenada en el dispositivo móvil información de contacto: nombre, teléfono, email, etc.

Cloud Computing: Se denominan servicios en la nube o cloud computing al modelo de servicios ofrecidos a través de la red, en el que los usuarios pueden acceder desde cualquier dispositivo con conexión a Internet, sin necesidad de almacenar la información de forma local.

CMT: Comisión del Mercado de las Telecomunicaciones. Es un Organismo Público regulador independiente de los mercados nacionales de comunicaciones electrónicas y de servicios audiovisuales.

Eclipse: Es un entorno de desarrollo integrado (IDE) open source compuesto de frameworks extensibles, herramientas y entornos de ejecución, para desarrollar, desplegar y gestionar el software en todo su ciclo de vida.

GPS: Global Positioning System. Sistema de posicionamiento global, es un sistema global de navegación por satélite (GNSS) que permite determinar en todo el mundo la posición de un objeto, una persona o un vehículo con una precisión hasta de centímetros (si se utiliza GPS diferencial), aunque lo habitual son unos pocos metros de precisión. El sistema fue desarrollado, instalado y actualmente operado por el Departamento de Defensa de los Estados Unidos.

J2EE: Java 2 Platform, Enterprise Edition. Es una plataforma de programación, parte de la Plataforma Java, para desarrollar y ejecutar software de aplicaciones en el lenguaje de programación Java con arquitectura de N capas distribuidas y que se apoya ampliamente en componentes de software modulares ejecutándose sobre un servidor de aplicaciones.

JSP: JavaServer Pages. Es una tecnología Java que permite generar contenido dinámico para web, en forma de documentos HTML, XML o de otro tipo. Las JSPs permiten la utilización de código Java mediante scripts. Además, es posible utilizar algunas acciones JSP predefinidas mediante etiquetas.

MAC: Media Access Control. La dirección MAC es un identificador de 48 bits (6 bloques hexadecimales) que corresponde de forma única a una tarjeta o dispositivo de red. Se conoce también como dirección física, y es única para cada dispositivo.

MVC: Modelo Vista Controlador. Patrón de arquitectura de aplicaciones software que apuesta por una separación de las distintas partes de la aplicación en el modelo que almacena los datos de la aplicación, la vista que interactúa con el usuario y el controlador que define la lógica de negocio.

Plugin: También llamado complemento, es una pequeña aplicación que se relaciona con otra para aportarle una función nueva y generalmente muy específica. Esta aplicación adicional es ejecutada por la aplicación principal e interactúan por medio de la API.

POJO: Acrónimo de Plain Old Java Object, es un término empleado en entornos Java para referirse a clases simples y que no dependen de un framework en especial, en particular para diferenciarlos de los EJBs o Enterprise Java Beans.

Push: Esta tecnología se refiere a un mecanismo de comunicación en el que la petición de una transacción se origina en el servidor a diferencia de los sistemas pull, en los que es el cliente el que solicita información al servidor.

RSSI: Receive Signal Strength Indication, Indicador de fuerza de señal de recepción. Este término se usa comúnmente para medir el nivel de potencia de las señales recibidas en las redes inalámbricas. Puede expresarse en Wattios o dBm.

SDK: Software Development Kit. Un kit de desarrollo de software es generalmente un conjunto de herramientas de desarrollo de software que le permite al programador crear aplicaciones para un sistema concreto. Puede ser tan sencillo como una API o puede, también, incluir hardware sofisticado para comunicarse con un determinado sistema embebido.

Servlets: Un servlet es un objeto que se ejecuta en un servidor o contenedor J2EE especialmente diseñado para ofrecer contenido dinámico desde un servidor web.El uso más común de los *servlets* es generar todas páginas web de forma dinámica a partir de los parámetros de la petición que envíe el navegador web.

Smartphone: Este término se refiere a los teléfonos inteligentes, teléfonos móviles con capacidades de computación y conectividad superiores a las convencionales, pudiendo remplazar a un ordenador personal en algunos casos. Una característica importante es que permiten instalar nuevas aplicaciones que amplíen sus funcionalidades.

SSID: Service Set IDentifier. Es un nombre incluido en todos los paquetes de una red inalámbrica (Wi-Fi) para identificarlos como parte de esa red. El código consiste en un máximo de 32 caracteres que la mayoría de las veces son alfanuméricos.

Usuario: Se refiere a la persona que dispone de un dispositivo móvil y tiene instalada y ejecutándose la aplicación.

XML: Extensible Markup Language es un formato de texto sencillo y muy flexible que desempeña un papel muy importante en el intercambio de datos en la Web. Es en realidad un metalenguaje, consiste en un formato para definir otros lenguajes por medio de etiquetas.

WIFI: WIreless FIdelity. WiFi es un mecanismo de conexión de dispositivos electrónicos de forma inalámbrica. Los dispositivos habilitados con WiFi, pueden conectarse a Internet a través de un punto de acceso de red inalámbrica. Existen diversos tipos de WiFi, basado cada uno de ellos en un estándar IEEE 802.11: IEEE 802.11b, IEEE 802.11g.

Capítulo 11: Bibliografía

Referencias empleadas en la elaboración del proyecto, junto con su autor y la fecha de la última consulta.

[1] CMT. 20/07/2012. http://www.cmt.es/c/document_library/get_file?uuid=7fd4c7c8-802c-4d10-bcd7-066b3933f91a&groupId=10138

[2] IIBA. 15/11/2011. http://www.theiiba.org/

[3] Apple. iOS. 20/11/2011. http://www.apple.com/es/ios/

[4] Google. Android. 20/11/2011. http://source.android.com/about/index.html

[5] Kantar Worldpanel. 15/06/2012. http://www.kantarworldpanel.com/dwl.php?sn=news_downloads&id=53

[6] Apache Software Foundation. The Apache HTTP Server Project. 03/12/2011 http://httpd.apache.org/

[7] Apache Software Foundation. Apache Tomcat. 03/12/2011. http://tomcat.apache.org/

[8] Google. Google App Engine. 03/12/2011 https://developers.google.com/appengine/

[9] Google. C2DM. 10/07/2012. https://developers.google.com/android/c2dm/

[10] Oracle. Netbeans. 03/12/2011. http://netbeans.org/

[11] The Eclipse Foundation. Eclipse. 03/12/2011. http://www.eclipse.org/

[12] Google. Google Play. 15/07/2012. https://play.google.com/intl/es/about/apps

[13] Foursquare. 05/12/2011. https://es.foursquare.com/

[14] Facebook. Facebook Places. 05/12/2011. https://www.facebook.com/about/location/

[15] Google. Google Latitude. 05/12/2011. http://www.google.com/intl/es_ALL/mobile/latitude/

[16] Ekahau. Ekahau. 07/02/2012. http://www.ekahau.com/

[17] Skyhook. Skyhook. 07/02/2012. http://www.skyhookwireless.com/

[18] Rosum Corporation. Rosum. 07/02/2012. www.rosum.com/

[19] Sonitor Technologies. Sonitor RTLS. 07/02/2012. http://www.sonitor.com/

[20] WifiSLAM. WifiSLAM. 07/02/2012. http://wifislam.com/

[21] Rong Peng and Mihail L. Sichitiu. "Angle of Arrival Localization for Wireless Sensor Networks," in Proc. of the Third Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks, (Reston, VA), Septiembre 2006.

[22] William Rison. “Time of Arrival Location Technique”. 2008. http://www.ee.nmt.edu/~rison/ee389_spr08/toa.pdf

[23] Mauricio Gende, Ivana Molina. “Trilatreración”. Junio 2011. http://catedras.fcaglp.unlp.edu.ar/geofisica/referenciacion-en-geofisica/teoria/instrumental-y-tecnicas-topograficas/trilateracion

[24] Trygve Reenskaug . Modells – Views – Controllers. Diciembre 1979. http://heim.ifi.uio.no/~trygver/1979/mvc-2/1979-12-MVC.pdf

[25] Google. WifiManager. Julio 2012. http://developer.android.com/reference/android/net/wifi/WifiManager.html

[26] Google. WifiInfo. Julio 2012. http://developer.android.com/reference/android/net/wifi/WifiInfo.html

[27] Samsung. Samsung Galaxy S. 15/02/2012. http://www.samsung.com/es/support/model/GT-I9000HKDVIP-techspecs

[28] Zyxel. Zyxel P660HWP-D1. 15/02/2012. http://www.zyxel.com/products_services/p_660hwp_dx_series.shtml?t=p

[29] U.S. Government. GPS Accuracy. 17/02/2012. http://www.gps.gov/systems/gps/performance/accuracy/

[30] The apache Software Foundation. Subversion. 22/03/2012. http://subversion.apache.org/

[31] Exentrique Solutions. XP-DEV. 22/03/2012. http://xp-dev.com/

[32] CollabNet. Subclipse. 22/03/2012. http://subclipse.tigris.org/

[33] Binghao Li, James Salter, Andrew G. Dempster and Chris Rizos. Indoor Positioning Techniques Based on Wireless LAN. 2006